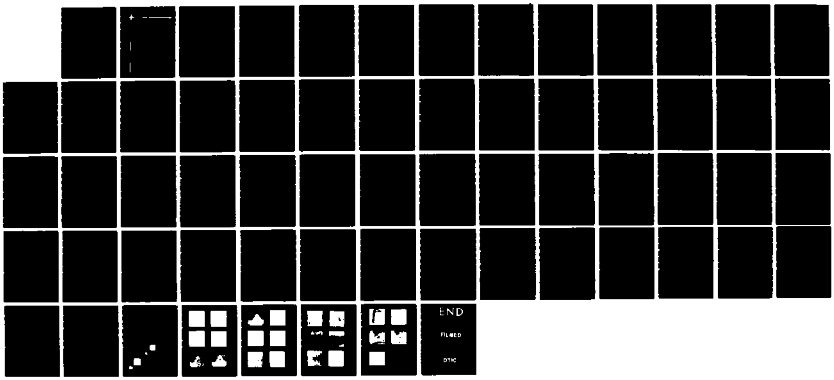
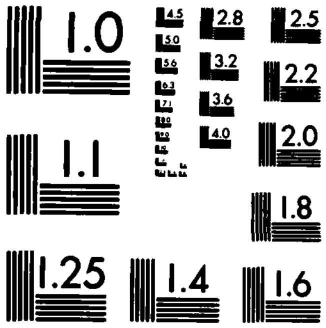
AD-A158 517 SIMPLE PARALLEL HIERARCHICAL AND RELAXATION ALGORITHMS FOR SEGMENTING NON..(U) BROWN UNIV PROVIDENCE RI LAB FOR ENGINEERING MAN/MACHINE SYST.. F S COHEN ET AL. 1/1

UNCLASSIFIED JUL 84 LEMS-7 ARO-18590.11-MA F/G 12/1 NL




1.0
4.5
2.8
2.5
5.0
5.6
3.2
2.2
6.3
7.1
3.6
8.0
4.0
2.0
1.1
1.8
1.25
1.4
1.6

ARO 18590.11-MA

Brown University

DIVISION OF ENGINEERING

PROVIDENCE, R.I. 02912

2

LABORATORY FOR ENGINEERING MAN/MACHINE SYSTEMS
LEMS

SIMPLE PARALLEL HIERARCHICAL AND RELAXATION ALGORITHMS FOR SEGMENTING NONCAUSAL MARKOVIAN RANDOM FIELDS

Fernand S. Cohen[1]
David B. Cooper[2]

Technical Report LEMS-7
July, 1984

2

LABORATORY FOR ENGINEERING MAN/MACHINE SYSTEMS
LEMS

SIMPLE PARALLEL HIERARCHICAL AND RELAXATION ALGORITHMS FOR SEGMENTING NONCAUSAL MARKOVIAN RANDOM FIELDS

Fernand S. Cohen[1]
David B. Cooper[2]

Technical Report LEMS-7
July, 1984

This work was partially supported by Army Research Office grant DAAG 29-81-K-0167, and National Science Foundation grants ECS-8119676 and ECS-8404774.

[1]Department of Electrical Engineering, Univ. of Rhode Island, Kingston, RI
[2]Division of Engineering, Brown University, Providence, RI 02912

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

AD-A158517

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER: ARO 18590.11-MA	2. GOVT ACCESSION NO.: N/A	3. RECIPIENT'S CATALOG NUMBER: N/A
4. TITLE (and Subtitle): Simple Parallel Hierarchical and Relaxation Algorithms for Segmenting Noncausal Markovian Random Fields		5. TYPE OF REPORT & PERIOD COVERED: Technical Report
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Fernand S. Cohen, David B. Cooper		8. CONTRACT OR GRANT NUMBER(s): DAAG29-81-K-0167
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Brown University, Providence, RI 02912		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS: U. S. Army Research Office, Post Office Box 12211, Research Triangle Park, NC 27709		12. REPORT DATE
		13. NUMBER OF PAGES
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report): Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report): Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report): NA		
18. SUPPLEMENTARY NOTES: The view, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number): Textured image segmentation; Markov Random Fields; Maximum likelihood segmentation; Image modelling; Adaptive image segmentation; Two dimensional Gaussian processes; Two dimensional binary processes; Texture model parameter estimation.		

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

The modelling and segmentation of images by MRF's (Markov Random Fields) is treated. These are two-dimensional noncausal Markovian Stochastic Processes. Two conceptually new algorithms are presented for segmenting textured images into regions in each of which the data is modelled as one of C MRF's. The algorithms are designed to operate in real

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

20. ABSTRACT CONTINUED:

time when implemented on new parallel computer architectures that can be built with present technology. A doubly stochastic representation is used in image modelling. Here, a Gaussian MRF is used to model textures in visible light and infrared images, and an auto-binary MRF to model a priori information about local image geometry. Image segmentation is realized as maximum likelihood estimation. In addition to providing a mathematically correct means for introducing geometric structure, the auto-binary (or ternary, etc.) MRF can be used in a generative mode to generate image geometries and artificial images, and such simulations constitute a very powerful tool for studying the effects of these models and the appropriate choice of model parameters. The first segmentation algorithm is hierarchical and uses a pyramid-like structure in new ways that exploit the mutual dependencies among disjoint pieces of a textured region. The second segmentation algorithm is a relaxation-type algorithm that arises naturally within the context of these noncausal MRF's. It is a simple, maximum likelihood estimator. The algorithms can be used separately or together.

Key Words

Textured image segmentation; Markov Random Fields; Maximum likelihood segmentation; Image modelling; Adaptive image segmentation; Two dimensional Gaussian processes; Two dimensional binary processes; Texture model parameter estimation.

Abstract

The modelling and segmentation of images by MRF's (Markov Random Fields) is treated. These are two-dimensional noncausal Markovian Stochastic Processes. Two conceptually new algorithms are presented for segmenting textured images into regions in each of which the data is modelled as one of C MRF's. The algorithms are designed to operate in real time when implemented on new parallel computer architectures that can be built with present technology. A doubly stochastic representation is used in image modelling. Here, a Gaussian MRF is used to model textures in visible light and infrared images, and an auto-binary MRF to model a priori information about local image geometry. Image segmentation is realized as maximum likelihood estimation. In addition to providing a mathematically correct means for introducing geometric structure, the auto-binary (or ternary, etc.) MRF can be used in a generative mode to generate image geometries and artificial images, and such simulations constitute a very powerful tool for studying the effects of these models and the appropriate choice of model parameters. The first segmentation algorithm is hierarchical and uses a pyramid-like structure in new ways that exploit the mutual dependencies among disjoint pieces of a textured region. The second segmentation algorithm is a relaxation-type algorithm that arises naturally within the context of these noncausal MRF's. It is a simple, maximum likelihood estimator. The algorithms can be used separately or together. The algorithms have the desirable properties: (i) the required computation appears to be close to the minimum required for segmenting images modelled by MRF's; (ii) the segmentation can operate in adaptive modes, estimating a priori unknown fixed or spatially varying MRF parameters; (iii) the segmentations are unusually accurate--- usually to within one or two pixels in the experiments run. A contribution of the paper is that a start is made in discussing and exploring that inherent structure of MRF's that must be exploited in order to construct physically meaningful MRF models for representing real textured images, and in order to devise segmentation algorithms that are computationally efficient.

1.0 History of the Textured Image Segmentation Problem

The use of Markovian Random Fields to model image textures, textured region geometry, and textured region boundary, and then the application of maximum-likelihood or Bayesian estimation methods for segmenting such images has a growing history. Cooper, et. al. [1], have used white Gaussian fields to represent texture, and unilateral (causal) Markov processes to represent region boundaries. Image segmentation was realized as the maximization of the conditional likelihood of a boundary given the data image. In [2], Cooper, et al., used the same representation and developed an approach that partitions an image into small square windows, does boundary estimation simultaneously in all windows through the use of dynamic programming, then seams the windows together using again dynamic programming. Again, using the same representation, Schenker, et. al. [3], demonstrated the hierarchical "ripple filter", a local relaxation-type algorithm to do maximum likelihood image segmentation. Elliott and Scharf [4] were the first to develop a dynamic programming algorithm for boundary estimation, and then Elliott, et.al. [5] developed a dynamic programming algorithm for segmenting multi-level texture. Here a white Gaussian field with different means (for the different textured regions) was used to model texture, and a MRF was used to model region geometry. This doubly-stochastic modelling for texture and region geometry was also adopted by Therrien [6]. He used a white or a causal colored texture field for describing the texture of the different regions in the image, and a binary field based on a 2D Markov chain introduced by Kaufman and Woods [7] to model prior information about texture region connectivity. The image segmentation algorithm he used was an iterative one where the texture region association of a pixel (i,j) was based on the relative numbers of pixels assigned to the 2 texture types in a small square about (i,j) and on its texture conditional likelihood. Later, maximum a posteriori estimation

(MAP) of an original image given the degraded observations through stochastic relaxation was also adopted by the Gemans [8]. They viewed the image as a pair of processes -- the intensity process and the line process. These were modelled by MRF's (with a relatively small neighborhood) whose Gibbs potential was assumed known. Image restoration was achieved through an annealing schedule that forces samples from the posterior distribution towards the minimal energy configuration (i.e. the maximum of the posterior distribution).

Finally, we mention three lines of research, not because they deal with the segmentation of MRF's, but rather because they use algorithms close in spirit to those that we discuss. Chen and Pavlidis [9] presented a hierarchical approach to textured image segmentation involving image data modelled by noncausal 2-D Gaussian processes. No use was made of MRF's or their properties. They expressed the segmentation problem as a sequence of tests of hypotheses on a quadtree data base and within the framework of a split-and-merge algorithm. Regions of arbitrary initial segmentation were tested for uniformity. If they were not uniform they were subdivided into smaller regions, or set aside if the appropriate statistics were below a given threshold. Subject to cluster analysis, similar uniform regions were merged as constituting a texture-type region. Updated estimates for the parameters of each random field were obtained as the uniform regions became larger. These were in turn used to classify some of the remaining unclassified small regions. Rosenfeld and colleagues have developed 'relaxation' algorithms for parallel processing. Though they do not do maximum likelihood estimation nor do they use the image data in iterations following the initial segmentation, nevertheless, they have shown that their approach involving "pseudo-likelihoods" ca used in many important applications, e.g., see [10].Faugeras and Berthod [11] devel a relaxation-like algorithm but based on the extremization of a performance functio Their application was object classification based on multi-spectral data. Hinton, et.al. [25], are developing elements of an approach to scene understanding based on networks, an energy performance functional , and a probabilistic relaxation optimization algorithm. Though the referenced papers do not deal with textured ima segmentation, there is a significant overlap in the kinds of equations that they and we ultimately deal with.

The first published papers on maximum likelihood segmentation of textured images when noncausal Gaussian MRF's are used for data modelling and another MRF is used for texture-type region modelling are those of Cohen, et al. [12, 13, 14]. Hierarchical and parallel iterative (relaxation-type) segmentation algorithms were introduced, and so was the use of these in combination.

1.1 Brief Overview of our Use of Windows, Our Approach to Textured Image Segmentation, and the History of Markov Random Fields for Texture Modelling.

We model textured images on two levels, one unobserved level to describe the geometry of the basic regions within each of which the image has one type of texture, and an observable level to describe the textured image data in each region. Under this scenario for image generation, nature first generates the texture-type regions, and then fills in each region with an image having the appropriate texture. Two-dimensional MRF's are used as texture and region models. The MRF models are a class of parametric models that have been studied extensively by Besag [15] and Bartlett [16] as a generalization to the Ising model. Gaussian MRF's have been extensively studied by Woods [17] and Jain [18], among others. Woods has considered them for image filtering [19]. They have been successfully used by Cross [20], and Kashyap and [21], among others, to model a variety of stationary texture fields such as Brodatz textures. Theory has been presented by Besag [15], as well as by Kashyap and Chelappa [21] for estimating parameters to adapt the models to stationary field data. Kashyap's and Chelappa's work [21] also included texture recognition for rectangular textured regions.

The segmentation problem consists of partitioning a textured image into regions, in each of which there is one of C possible texture types (texture classes). This means that a texture-type classification must be made for each pixel. Since a connected single texture-type region will usually contain many pixels, we consider a simplification that considerably reduces the required computation. The simplification is to partition the image into small square windows and to process each window under the assumption that it contains one or at most two texture types. Then each window is processed as a separate subimage, and the results combined in an appropriate way (see Section 4.5). Hence, most of this paper is concerned with segmenting a small square window of image into two regions, each of which comprises one of two image texture types. Extension to three or more texture types in a window is immediate.

A precise brief statement of our segmentation approach is the following. Nature partitions an NxN pixel window into two (or more) regions denoted as texture-type regions 0 and 1. Within region k, model k is used to generate the data. The image data in region 0 is statistically independent of that in region 1. Let s_{ij} be a binary variable taking values 0 and 1, denoting texture-type 0 and 1, respectively, and let S denote the N^2 dimensional vector having components s_{ij}. Then the vector S specifies the partition of the NxN pixel window into the texture-type 0 and texture-type 1 regions. Let y_{ij} denote the picture function (i.e., image data) at pixel (i,j), and Y be the N^2 component vector having components y_{ij}. Then Y is the vector of all picture function values in the NxN pixel window. If S is viewed as a constant but unknown vector, the likelihood of the data in the window is

$$p(Y,S) = p(Y_0|0)\ p(Y_1|1) \qquad (1)$$

Here, p(Y,S) is the liklihood of Y given the partition S. Y_k denotes the picture function vector for texture-type region k, and $p(Y_k|k)$ denotes the likelihood of Y_k given the probability model for image data in texture-type region k. Hence, the simplest texture segmentation problem is to determine the partition S for which Eq.1 is a maximum.

The second problem is that in which S is itself a random vector. That is, nature partitions the window into texture-type regions in accordance with some probabilistic process. Then the segmentation problem is the determination of S for which the posterior likelihood

$$p(S|Y) \tag{2}$$

is a maximum. Note, equivalent to maximizing (2) is finding the S that maximizes (3) or (4)

$$\ell n\ p(Y,S) \tag{3}$$

$$\ell n\ p(S) + \ell n\ p(Y|S) \tag{4}$$

1.2 Contribution of this Paper

The present paper is an in-depth treatment and extension of material briefly introduced in [13,14]. It makes the following contributions. Certain properties of MRF's need to be exploited in order to do physically meaningful image data modelling that is also mathematically consistent. Some of these questions are raised and one solution is presented. Parallel iterative and hierarchical algorithms are presented for maximum likelihood segmentation of textured images. Our iterative algorithm will generally require much less computation than will an annealing algorithm, and will also work with nonstationary models and data, whereas the annealing algorithm usually will not. Again, there are properties of the MRF's that can be exploited, this time to substantially reduce the amount of required computation. A divide-and-conquer approach is taken in our hierarchical segmentation algorithm that reduces the number of multiplication-equivalent operations to 42,257 for a 64x64 pixel window. This is to be compared with ten to fifteen times that number of multiplication-equivalent operations for a direct hierarchical approach. The simplest iterative algorithm is mathematically correct, but lacks a certain physical meaningfulness. This problem is addressed and one solution is briefly proposed. A number of experiments are run with real data. This turns out to be important, as the segmentation of real data is much more difficult than is that of artificially generated data---- especially since real image texture data is generally nonstationary. One approach to parameter-adaptive segmentation is briefly discussed and illustrated. This estimation of unknown model parameters during the segmentation process is required in practice. Our algorithms appear to be robust, and effective even when the models do not represent the data well.

2.0 Markov Random Fields (MRF)

We use the auto-normal MRF (i.e., Gaussian process) as the texture data model, and the auto-binary process for modelling region geometry for an image window that contains at most two different texture types. Let r=(i,j) index pixel location where i,j specify pixel row and column location and satisfy $1 \leq i,j \leq N$. Let x_r denote a random field, with x_r the field at pixel r, X a vector specifying the field over an entire N x N window and having components x_r, and X_r the field everywhere but at pixel r. By $\{x_r\}$ a MRF we mean that

$$p(x_r|X_r) = p(x_r|x_v, v\varepsilon D_p)$$

Here, D_p denotes a neighbor set, and $p(x_r|X_r)$ denotes the conditional likelihood of x_r given X_r.

The nature of D_p is illustrated in Fig. 1. As indicated in Fig. 1, a 1st order MRF is one for which the neighbor set consists of the four pixels constituting the north, south, east, and west neighbors of the center pixel. A

2nd order MRF is that for which the neighbor set is the first layer of eight pixels surrounding the center pixel, i.e., all pixels marked 1 or 2. A 3rd order neighbor set consists of all pixels marked 1, 2 or 3, etc. In general,

$$D_P = \{v = (\ell, m) \text{ such that } ||r-v||^2 \leq N_p \text{ and } v \neq r\}$$

Where P is the order of the process, and N_p is an increasing function of P. N_p takes the values 1,2,4,5,8,9 for P=1,2,..,6 respectively.

<u>2.1 The Gaussian MRF Texture Model</u>

The observed texture field (i.e., image data field) is denoted y_{ij} or just y_r, $r = (i,j)$. The conditional likelihood for the kth texture class is Gaussian, and is given by

$$p(y_r \mid Y_r, \text{ class } k) = [2\pi\sigma^2(k)]^{-\frac{1}{2}} \exp\{-(1/2\sigma^2(k))[y_r - \mu(k) - \sum_{v \in D_p} \beta_{r-v}(k)(y_v - \mu(k))]^2\} \qquad (5)$$

For $\{y_r\}$ stationary, β_{r-v} must satisfy certain restrictions. However, nonstationary fields are also of interest in this paper. For a stationary field, the joint probability density function (pdf) for the image in texture region k given that the field is 0 outside this region is shown by Besag [15] to be

$$p(Y|\text{class } k) = [2\pi\sigma^2(k)]^{-M/2}|B(k)|^{1/2}\exp\{-[(Y-U(k)]^t B(k)[Y-U(k)]/2\sigma^2(k)\} \qquad (5a)$$

where $U(k) = \mu(k)(1,1,..,1)^t$, and Y is an (M x 1) column vector whose elements are the picture function y_r at the pixels in the M pixel region. B(k) is a symmetric positive definite (MxM) matrix whose diagonal elements are unity, and whose off-diagonal element is equal to $-\beta_{r-v}$, where β_{r-v} indicates the strength of the spatial interaction between the MRF at points r and v. β_{r-v} is zero if points r and v are not neighbors.

2.2 The Binary MRF Spatial Region Model

We use the noncausal binary process to model region geometry. It has been used by others to model image textures (Cross [20]). For example, if one texture-type region consists of small elongated blobs and another consists of large regions, a binary MRF model can be designed to generate such patterns. Or if two texture-type regions are large with boundaries of low curvature, a binary model can be designed to generate these patterns. For the local 0,1 pattern shown in Fig. 2, if nature's goal was to generate regions with long smooth boundaries she would generate an s value of 0 with higher probability than an s value of 1 at location x. Let S denote the region-field modelled by the noncausal auto-binary Markov field that describes the geometry of the regions in the image. For an N x N image, S has N^2 components $\{s_{ij}\}$, each s_r taking the value 0 or 1 and describing the allocation of pixel r = (i,j) to either class I or class II. The conditional probability takes the form

$$p(s_r|S_r) = \exp(s_r T_r)/[1 + \exp(T_r)] \tag{6}$$

where T_r is given by

$$T_r = a + \sum_{v \in D_p} b_{rv} s_v \tag{7}$$

For $\{s_{ij}\}$ stationary, $b_{rv} = b_{r-v}$ for all r = (i,j) and v = (m,n).

3.0 The Segmentation Problem

Overview

The window segmentation problem of concern in this paper is that a window may contain two* texture-type regions, each from one of C classes, and the window is to be partitioned into two regions, each consisting of one texture

* Three or more are possible, but the required computation is more extensive.

type. We use maximum likelihood estimation for this purpose. The two difficulties that arise in the segmentation problem are: first, likelihoods must be computed for texture regions of irregular shape; second, likelihoods must be computed for many of the possible partitionings of the window. The challenge of the first problem is that there is no obvious simple way of computing the joint likelihood of a noncausal MRF over an irregular region. The challenge of the second problem is to be able to compute the picture function likelihoods for many different partitions of a window without having to do a horrendous amount of computation.

This paper presents simple practical solutions to both problems. The first algorithm we present is a hierarchical pyramidal-type algorithm, whereas the second one is an iterative relaxation-type algorithm. Both algorithms can handle textured images modelled by either stationary or nonstationary MRF texture models.

The reason for partitioning the image into windows is threefold: 1) it permits parallel processing, since windows can be segmented simultaneously using appropriate hardware architectures, and the results then seamed together; 2) by having one or at most two texture classes in a window, the processing is relatively simple; 3) if the windows are small, the texture model within a window can be treated as being spatially stationary, i.e., having parameters that are constant for the entire window (and satisfying certain restrictions). This is especially convenient if, as is usually the case, the texture model parameters are a priori partially unknown or are spatially slowly varying and must be estimated during the segmentation process.

3.1 The Hierarchical Algorithm.

In the hierarchical algorithm, we often refer to the two texture-type regions as object and background regions. The reason for this is partly historical and stems from the fact that a goal is often to isolate a specific object without caring about the interpretation of the surrounding region. Examples are an image of a kidney in a CATSCAN, a vehicle in an infrared image, a tree in a visible light image, etc. Such an object often appears in an image as a convex textured blob. The intersection of a window with such a region will be connected. Hence, for the hierarchical algorithm we have imposed the constraint that the estimate of one of the textured regions in a window, the object region, be connected. Our algorithm can easily be modified to require that the estimated object and background regions both be connected regions, or to not require connectivity in either type of region estimate.

The segmentation sought is that which maximizes the likelihood of the data. In each window the segmentation algorithm is hierarchical and uses a quadtree-like data structure. The window is divided into four quadrants and the best segmentation, i.e, grouping of these four blocks, is obtained. Then each of the four blocks is itself partitioned into four, and the process is repeated until the block size is such that each block consists of one pixel. At each stage of the hierarchy, the object connectivity constraint is met.

Each stage of the algorithm involves working with blocks of a certain size and consists of two steps.

Step 1. "Region growing".

At the end of the nth stage, the window has been segmented into two sets, one of which is referred to as the object region and is constrained to be connected. For the (n+1)st stage, we shrink and expand the object by partitioning each block in the neighborhood of the estimated boundary into four and obtain the best grouping of the four resulting quadrants. There will be 16 partitions to consider. Since under the auto-normal model assumption neighboring blocks are spatially interacting among each other (i.e., partially correlated along their common boundaries) the likelihood of each partition is conditioned on the segmentation of the surrounding blocks determined at the nth stage. This use of stochastic dependence is crucial to achieving good segmentation. This is especially true when the blocks are of small sizes, since a small block may contain only a portion of a cycle of texture data, and consequently, texture-type classification cannot be based on the data content of the small window alone. Rather, use must be made of the continuity of the texture data and its first or higher derivatives at the boundary between the small window and the surrounding image region of the same texture type. Our maximum likelihood segmentation algorithms automatically include information roughly equivalent to this. All object blocks constituting the boundary of the object as well as all background blocks neighboring the boundary blocks are processed simultaneously.

Step 2. "Object connectivity"

We check for the single-object-region connectivity constraint. If there is more than one object-region, we merge them into one. The merging is done by computing the likelihoods of all possible configurations that will result in a single connected object-region. We have adopted the notion that no pair of regions be merged unless they have at least one common first-neighbor block. By first neighbor blocks we mean background blocks that are adjacent to the boundary of the object. We repeat step 1 and step 2 at each stage of decreasing block size until each block consists of one pixel.

The following example will illustrate the algorithm steps more clearly. Suppose at the end of the 2nd stage we obtained the segmentation shown in

Figure 3a. Note that the object is not connected, as it consists of two disjoint regions R1, R2. Following step 2 there will be one of two possible results. Either block B is reallocated to the object class, figure 3b, or R1 or R2 are reallocated to the background class, figure 3c,d, whichever is more likely. This results in one connected object region R. We expand and shrink region R by partitioning the boundary and first-neighbor blocks as in step 1.

3.2.1 Computational Considerations for Stationary Gaussian MRF Textures

The hierarchical texture segmentation algorithm involves recognizing the texture region association of large blocks of image data in the early stages and small blocks of image data in the last stages of image segmentation. For the former, it is necessary to compute the likelihoods of large blocks of the image. For the latter, it is necessary to compute the likelihoods of small blocks of data conditioned on their surrounds. These are normally computationally prohibitive tasks. In this section we will show how to practically compute the likelihoods of large blocks and the conditional likelihoods of small blocks of data conditioned on their surrounds.

The free-boundary condition [15] is assumed. The meaning of this is that if pixel (i,j) is in the region for which the joint likelihood of all picture function values is to be computed and if pixel (ℓ,m) is not in the region but is in the conditioning neighbor set given in (5), the picture function $y_{\ell,m}$ at (ℓ,m) is not to be used in (5). (Note, (ℓ,m) is in the conditioning neighbor set if $(\ell,m) \in D_p$.) Not including $y_{\ell,m}$ is equivalent to setting it to 0 in (5). The use of this free boundary condition does not give consistent probabilities for all subsets within the region. We comment on this briefly in Section 7.0. For an M x N rectangular lattice, the joint density function is given in (8). For simplicity, we assume that $\bar{Y}$ ($\bar{Y}$=Y-U(k)) represents the picture function minus its mean. $\bar{Y}$ is a vector with components $\bar{y}_{ij}=y_{ij}-u(k)$ that are arranged in raster scan order.

$$p(Y) = (2\pi\sigma^2)^{-(MN)/2} |B|^{\frac{1}{2}} \exp\{-\bar{Y}^t B \bar{Y}/2\sigma^2\} \tag{8}$$

As M,N $\to\infty$, |B|, the determinant of B, is asymptotically equal to the determinant resulting from the use of a torus structure [21]. (This equality is in the sense that the difference of the two determinants divided by either one of them goes to 0.) This occurs because as M,N $\to\infty$, the contribution to p(Y) of the y_{ij} near the window border becomes unimportant; hence the choice of boundary condition becomes unimportant. For the torus structure, |B| takes the simple form [21]

$$|B| = \prod_{i=0}^{M-1} \prod_{j=0}^{N-1} [1 - 2 \sum_{-2\le \ell,m \le 2} \beta_{\ell m} \cos\left(\frac{2\pi}{M} \ell i + \frac{2\pi}{N} mj\right)] \tag{8a}$$

Because of the raster scan, the MN x MN positive definite symmetric matrix B can be decomposed as

$$B = \begin{bmatrix} B_1 & B_2 & B_3 & 0 & \cdots & & 0 \\ B_2^t & B_1 & B_2 & B_3 & & & \\ B_3^t & B_2^t & B_1 & B_2 & & & \\ \cdot & & & & & & \\ \cdot & & & & & & \\ \cdot & & & & & & B_3 \\ & & & & & & B_2 \\ 0 & 0 & \cdot & \cdot & B_3^t & B_2^t & B_1 \end{bmatrix} \tag{8b}$$

B_1, B_2 and B_3 are (M X M) matrices. The matrix B_1 represents the spatial interaction of a row (column) with itself, while B_2 is the spatial interaction matrix between two adjacent rows (columns), and B_3 is the spatial interaction matrix between two rows (columns) separated by one intervening row (column). Then for a 2nd order model the quadratic in (8) is expanded as

$$\bar{Y}^t B \bar{Y} = \sum_{i=1}^{M} \sum_{j=1}^{N} \bar{y}_{ij}^2 - 2\beta_{11} \sum_{i=1}^{M-1} \sum_{j=1}^{N} \bar{y}_{ij}\bar{y}_{i+1,j} - 2\beta_{12} \sum_{i=1}^{M} \sum_{j=1}^{N-1} \bar{y}_{ij}\bar{y}_{i,j+1}$$
$$- 2\beta_{21} \sum_{i=1}^{M-1} \sum_{j=1}^{N-1} \bar{y}_{ij}\bar{y}_{i+1,j+1} - 2\beta_{22} \sum_{i=1}^{M-1} \sum_{j=2}^{N} \bar{y}_{ij}\bar{y}_{i+1,j-1} \quad (9)$$

We proceed now to show how to recursively compute the conditional and unconditional likelihood functions associated with the 16 partitions of a boundary or first neighbor block.

<u>Result 1</u>

Suppose an N X N data block Y is divided into 4 quadrants. Let Y_j be the data vector in quadrant j (j = 1,2,3,4) where these quadrants are NW (Northwest), NE, SW and SE, respectively. Then the $N^2 x N^2$ B matrix associated with the data block Y can be partitioned into

$$B = \begin{bmatrix} B_1 & B_{12} & B_{13} & B_{14} \\ B_{12}^t & B_2 & B_{23} & B_{24} \\ B_{13}^t & B_{23}^t & B_3 & B_{34} \\ B_{14}^t & B_{24}^t & B_{34}^t & B_4 \end{bmatrix}$$

where $B_j (j = 1,2,3,4)$ is the interaction matrix associated with Y_j and is given in (8b) for $M = N = N/2$. B_{ij} is the interaction matrix between quadrants i and j. Note that the B_j are the same for all quadrants, and that $B_{12} = B_{34}$ and $B_{13} = B_{24}$. The joint likelihood of the N X N data block Y can then be expressed as

$$p(Y) = (2\pi\sigma^2)^{-N^2/2}|B|^{1/2}\exp\{-(1/2\sigma^2)[\sum_{j=1}^{4} \bar{Y}_j^t B_j \bar{Y}_j + 2 \sum_{\substack{i,j \\ 1\leq i<j\leq 4}} \bar{Y}_i^t B_{ij} \bar{Y}_j]\} \quad (10)$$

Proof follows from the preceding decomposition of B.

Let $I_W(j)$ be the within-block interaction term for data block Y_j. $I_W(j) = \bar{Y}_j^t B_j \bar{Y}_j$. Let $I_B(i , j)$ be the between-block interaction term for blocks Y_i and Y_j. $I_B(i , j) = \bar{Y}_i^t B_{ij} \bar{Y}_j = \bar{Y}_j^t B_{ij}^t \bar{Y}_i$. Hence

$$\ell n\ p(Y) = (1/2)\ell n|B| - (N^2/2)\ell n\ 2\pi\sigma^2 - 1/2\sigma^2[\sum_{i=1}^{4} I_W(i) + 2 \sum_{\substack{i,j \\ 1\leq i<j\leq 4}} I_B(i,j)]$$

The next result gives the form of the likelihood of a block X conditioned on the surrounding block(s) Y.

Result 2

The conditional likelihood of a data block X given surrounding data blocks Y and the free boundary condition outside blocks X and Y is

$$p(X|Y) = (2\pi\sigma^2)^{-N_X/2} |B_X|^{\frac{1}{2}} \exp\{-(1/2\sigma^2)[\bar{X}^t B_X \bar{X} + 2\bar{X}^t B_{XY}\bar{Y} + \bar{Y}^t B_{XY}^t B_X^{-1} B_{XY}\bar{Y}]\} \quad (11)$$

where N_X is the number of pixels in block X.

Proof: See Appendix A

Remark

~~As $N_X \to \infty$, $Y^t B^t_{XY} B_X^{-1} B_{XY} Y / X^t B_X X \to 0$. Hence, for large N_X and low-order process $Y^t B^t_{XY} B_X^{-1} B_{XY} Y$ can be neglected and we use the approximation~~ (11) can be computed exactly. In our experiments we have used

$$\ell n p(X|Y) = (\tfrac{1}{2})\ell n|B_X| - (N_X/2)\ell n(2\pi\sigma^2) - (1/2\sigma^2)[I_W(X) + 2I_B(X,Y)]$$

with good results.

From Result 2, it follows that the conditional likelihoods involved in the decision mechanism are explicitly expressed as a summation of appropriate within-block and between-block interactions. These terms can be recursively computed from one resolution to the next. For example, assume that at stage n the window has M^2 blocks. Let $I^c_W(i,j;n)$ be the within-block interaction for the (i,j) block at resolution n under texture class c. Let $I^c{}_B(i,j,\ell,m;n)$ be the between-block interaction between block (i,j) and block (ℓ,m) at resolution n under texture class c. (Note that increasing n means smaller blocks, with the largest n specifying a block size of one pixel). Then the within-block interactions at resolution (n - 1) are computed from $I^c_W(.,\ .;\ n)$ and $I^c_B(.,.,.,.;n)$ using a very simple ring-structure shown in figure 4. Here the $I_W(.,.;n)$ for each block at resolution n is represented by a node, whereas the $I_B(.,.,.,.;n)$ between a pair of blocks is represented by either a dashed or solid branch. We assume that the sets $I_W(.,.;n)$ and $I_B(.,.,.,.;n)$ for the blocks at resolution n have been computed and stored. We compute the sets $I_W(.,.,;n-1)$ and $I_B(.,.,.,.,;n-1)$by summing up the branches of the appropriate solid rings, and the branches of the appropriate dashed rings given at resolution n (see fig. 4). Note that the between-block interaction between 2 non-neighboring blocks is zero. Here the neighborhoood set is that which is defined by the order of the process. For a 5th-order process the only nonzero interaction blocks that a block (i,j) of size (2 x 2) or higher has is the between block interaction set $\{I_B(i,j,i+k,j+\ell)$, $-1 \leq k,\ \ell \leq 1$, and

$(k,\ell) \neq (0,0)\}$. With special-purpose hardware one can compute all the $I^c_W(.,.;n-1)$ and $I^c_B(.,.,.,.;n-1)$ simultaneously using $(I^c_W(.,.;n)$, $I^c_B(.,.,.,.;n-1))$. Because of this recursion, the hierarchical algorithm is computationally attractive.

3.2.2 <u>Amount of Computation Required for Hierarchical Segmentation Using Stationary Gaussian MRF Textures</u>

We can estimate the amount of required computation for a 64 x 64 window and 2nd-order model as follows. At the 2 x 2 block level, 14 multiplications and 9 additions are required for computing the within-block interaction term for each block, and there are $(32)^2$ such blocks. Hence a total of $(32)^2$ x 14 multiplications and $(32)^2$ x 9 additions are required. For the between-block interaction of a block with any one of its neighbors, 7 multiplications and 3 additions are needed for the block. The total number of multiplications for the between-block interactions is 7[(2x29x30+2x29x29)+4x30+4x2x29+29+4x3] = 7x3786 multiplications and 3[(2x29x30+2x29x29)+4x30+4x2x29+4x3] = 3x3786 additions. Hence, at the (2 x 2) block resolution, $(32)^2$x14+7x3786 = 40,838 and 3x3786 = 20,574 additions are used. For resolution (n-2), each block is (4 x 4). Using the ring structure, 9 additions and one multiplication are needed for the within-block interaction for each block, hence a total of $(16)^2$ multiplications and $(16)^2$ x 9 additions. For the between-block interaction between two (4 x 4) blocks, we need 1 multiplication and 6 additions, hence a total of [(2x13x14+2x13x13)+(14x4+2x13x4)+4x3)] = 882 multiplications and 6x882 additions, etc....

Total number of multiplications for all the $I_W(.,.;.)$ is 14,677.

Total number of additions for all the $I_W(.,.;.)$ is 9556.

Total number of multiplications for all the $I_B(.,.,.,.;.)$ is 27,580.

Total number of additions for all the $I_B(.,.,.,.;.)$ is 27,042.

Without this ring structure, the total number of multiplications for $I_W(.,.;.)$ is <u>107,902</u> multiplications and for $I_B(.,.,.,.;.)$ is <u>44,020</u> -- hence, a total of <u>151,922</u> multiplications compared to <u>42,257</u> multiplications. Furthermore, if the "within" and the "between" block structures that we developed here are not used, but rather p(Y,S) is computed from scratch for each possible partition in the hierarchical algorithm, then the number of multiplication equivalent operations for segmentation is at least fifteen to twenty times the 42,257 required for our algorithm.

3.2.3 Amount of Required Computation with a Parallel Architecture

The bulk of the computation in 3.2.2 could be carried in parallel with a <u>simple</u> special purpose architecture. Here the overall structure is such that each block at a given resolution (say (n-1)st) is assigned nine processors. The first one, called the within-processor, computes the $I_W(i,j;n-1)$ term for the (i,j) block at the (n-1)st resolution. It has as inputs the elements of the appropriate solid ring at resolution n. An example is shown in Fig 4. The remaining eight processors are called the between-processors. Since each block of size (2x2) or higher has eight neighbors as shown in Fig 5 for a MRF of 2nd order through fifth order, each between processor has the task of computing an $I_B(i,j,.,.;n-1)$ term for the interaction between block (i,j) and one of its 8 neighbors. Each between-processor has as inputs the elements that constitute the appropriate dashed ring at resolution n. Again an example is shown in Fig. 4. At the highest resolution (i.e. at 2x2 block level) the total number of multiplications, and additions is 14 and 9, respectively, for the within-processor, and a maximum of 4 for the between processor for a 2nd order MRF (for a 5th order that number will be 12). For lower resolutions only 1 multiplication and 9 additions are needed for $I_W(.,.;.)$. A maximum of 1 multiplication and 4 additions are required for $I_B(.,.,.,.;.)$. The total computation for $\{I_W(.,.;.)\}$ is $14 + \log(N-1)$ multiplications and $9 \log N$ additions; and a maximum of $4+\log(N-1)$ multiplications and $4\log N$ additions for $I_B(.,.,.,.;,)$. An upper bound for the required computation is then $14+\log(N-1)$ multiplications and $9\log N$ additions. Note that except at the 2x2 block level, the processors consist mainly of adders and are hence structurally simple!

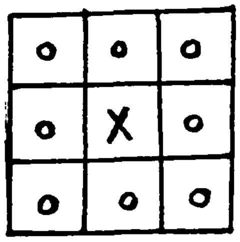


Figure 5

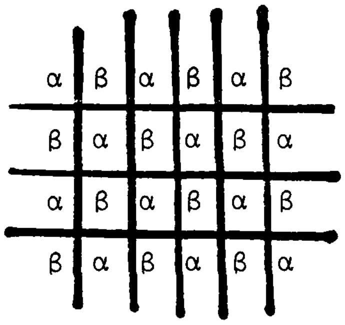


Figure 6

3.3.0 Pseudo-Likelihood

Let Y_{ij} denote the vector Y less the component y_{ij}. Consider $y_{ij}|Y_{ij}$ (i.e., y_{ij} conditioned on its surround), and suppose the window is just one texture type. For simplicity, assume the texture model is a 1st order Gaussian MRF. If the y_{ij} are limited to one code, i,e., the pixels marked α or those marked β in Fig. 6, the conditioned variables $y_{ij}|Y_{ij}$ will be statistically independent. Hence,

$$\prod_{\substack{(i,j) \\ \varepsilon \text{ one code}}} p(y_{ij}|Y_{ij}) \tag{12}$$

is the joint conditional likelihood of picture function values for pixels in one code. Equivalently, (12) is the joint likelihood of the statistically independent residuals in the prediction of the y_{ij} in a code given the surrounding picture function values. By the pseudo-likelihood of the picture function in a region, we mean the product of the conditional likelihoods of the picture function at all points in the region, e.g., (12a). Note that the pseudo-likelihood is not a true likelihood; rather it is the product of joint conditional likelihoods, each joint conditional likelihood being for one code in the region.

$$\prod_{\substack{(i,j)\\ \epsilon\ \text{region}}} p(y_{ij}|Y_{ij}) \tag{12a}$$

Let the window be NXN. Denote by

$$p(y_{ij}|Y_{ij},\Phi) \tag{13}$$

the conditional p.d.f. of y_{ij} given its surround when the p.d.f. parameters are given by the vector Φ. Then it is easy to show that for the field y_{ij} having parameter vector Φ_t with $\Phi_t \neq \Phi$,

$$P\{\lim_{N\to\infty}[\prod_{(i,j)} p(y_{ij}|Y_{ij},\Phi_t)/p(y_{ij}|Y_{ij}\ \Phi)] > 1\} = 1 \tag{14}$$

That is, the ratio of the conditional likelihoods of the picture function values in a code becomes greater than 1 for almost all windows as $N \to \infty$. Hence, this conditional likelihood ratio is an effective texture-type classifier. Furthermore, it is trivial to show (Appendix B) (also [21]) that (14) holds if all pixels in a window are used, and better performance is to be expected then. We suggest using this pseudo-likelihood function as a performance functional for segmenting MRF's when the pseudo-likelihood estimated parameters for the Gaussian MRF are those for a nonstationary process.

3.3.1 On the Use of the Pseudo-Likelihood Function and Maximum Pseudo-Likehood Parameter Estimates

Though we successfully perform adaptive maximum-likelihood textured-image segmentation, two problems are encountered. The first is the computation of determinants such as those in (5a) for irregularly shaped regions. Approximations are necessary here. In our hierarchical algorithm we use (8a) for computing $|B|$. The approximation is satisfactory here because the algorithm considers only rectangular regions, square regions, or a combination of both as shown in Fig. 7. For the latter case, we use the approximation $|B| \approx |B_r||B_s|$. For an arbitrary shaped region the approximation in (8a) may not be

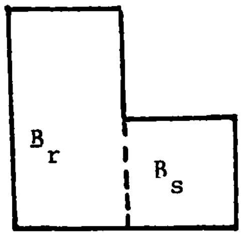


Figure 7

appropriate. The second more serious problem is that computation-intensive maximum likelihood parameter estimation is necessary if the MRF models used are constrained to be those for stationary processes. Using the asymptotic expression for the likelihood (5a), good parameter estimates can be obtained for a stationary Gaussian MRF model even when the data is somewhat nonstationary. However, because of the large amount of required computation, online adaptive segmentation may not be possible. On the other hand, the maximum pseudo-likelihood parameter estimates for the stationary MRF are asymptotically consistent, efficient, and computationally simple [21] --- even for highly irregularly shaped regions. (However, there is no guarantee that the estimates based on finite data sets will be those for a stationary MRF, nor even those for a Gaussian MRF, i.e., the estimated parameters may result in a determinant in (5a) that is not positive semi-definite.) The pseudo-likelihood function can also be used to estimate parameters for nonstationary Gaussian MRF's, or for MRF's that are not Gaussian. Fortunately, we have found generally that these pseudo-likelihood estimates of the parameters in (5) can be used in a pseudo-likelihood performance functional (Sec. 3.3.0, 6.0) that is effective for both texture recognition and hierarchical textured image segmentation. Hence, when the pseudo-likelihood estimates for the β_{ij} result in positive-semidefinite B's for (5a), our hierarchical maximum-likelihood segmentation can be used, and when this estimated B is not positive-semidefinite, maximum pseudo-likelihood segmentation can be used. One other option for hierarchical segmentation is the use of the likelihood performance functional --- but neglecting the determinant of B in (5a) when the estimated B is not positive-semidefinite. Experiments using these three performance functionals are described in Section 6.0.

3.3.2 Amount of Computation Required for Hierarchical Segmentation When Using Nonstationary MRF Textures

Similar to the likelihood function of a Gaussian MRF, the quadratic of the pseudo-likelihood function can be expressed as sums of data covariances. The neighborhood structure is larger, however. This can be seen by considering a 1st order Gaussian MRF. The quadratic in the conditional function $p(y_{ij}|Y_{ij})$, given by

$$[\bar{y}_{ij} - \beta_{11}(\bar{y}_{i-1,j} + \bar{y}_{i+1,j}) - \beta_{12}(\bar{y}_{i,j-1} + \bar{y}_{i,j+1})]^2/2\sigma^2,$$

involves covariances (interaction terms) encountered in a 3rd order MRF process (e.g. $\bar{y}_{i-1,j}\ \bar{y}_{i+1,j}$, $\bar{y}_{i-1,j}\ \bar{Y}_{i,j+1}$, etc.) The pseudo-likelihood function includes covariances in the β_{21}, β_{22}, β_{31}, β_{32} directions which weren't present in the likelihood function. The quadratic function in the pseudo-likelihood for a 1st order MRF can be shown to be

$$c_{oo}\sum_{i,j}\bar{y}^2_{ij} + c_{11}\sum_{i,j}\bar{y}_{ij}\bar{y}_{i+1,j} + c_{12}\sum\bar{y}_{ij}\bar{y}_{i,j+1} + c_{21}\sum\bar{y}_{ij}\bar{y}_{i+1,j+1}$$
$$+ c_{22}\sum_{i,j}\bar{y}_{ij}\bar{y}_{i+1,j-1} + c_{31}\sum_{i,j}\bar{y}_{ij}\bar{y}_{i+2,j} + c_{32}\sum_{i,j}\bar{y}_{ij}\bar{y}_{i,j+2}$$
+ correction.

Hence, a similar ring-structure is appropriate to evaluate the $\{I_W(.,.;.)\}$ and $\{I_B(.,.,.,.;.)\}$ sets. The amount of computation is higher than that for the likelihood function, as a larger number of covariance functions are involved, and there is a correction term which is nonexistent when the likelihood function is used.

3.3.4 Generality of the ring-structure

The ring structure is fundamental to MRF's in genral, not just the Gaussian. For the class of auto-models [15], the most general form p(Y) can take in order to give a valid probability sturcture that represents a MRF is given by

$$p(Y) = h \exp Q(Y) \qquad (15)$$

where h, the partition function, is a normalizing constant in order that p(Y) have unit area. Q(Y) is known as a Gibbs potential and is shown [15] to have the form

$$Q(Y) = \sum_{r} y_r \, G_r \, (y_r) + \sum_{r,v} G_{r-v}(y_r, y_v) \; y_r y_v \qquad (16)$$

This has the same form as the quadratic for the Gaussian MRF and therefore the compuation involved in the hierarchical segmentation can be carried on through a similar ring-structure. This fact makes the hierarchical segmentation computationally attractive and highly parallelizable for any random field for which the Gibbs potential has been determined.

4.1. Segmentation Using Auto-Binary Fields for Region Geometry and a Parallel Iterative Relaxation-Type Algorithm

In Section 3, the segmentation of an image into regions each of which contains a single Markovian texture field was considered. The hierarchical algorithm was based solely on models for the observable texture fields, and did not incorporate any information about image geometry (i.e., shapes of the regions, their sizes, etc.). Because of the lack of information about image-geometry, boundaries between segmented regions in noisy images are often jagged and must be smoothed. A noncausal auto-binary field could be used towards that end. The auto-binary field is used to model region geometry or boundary shape.

For the case of an object that is assumed to have a boundary that is smooth and slowly curving, an appropriate field will be such that the conditional probability of a point s_{ij} given its neighbors will be low for the case where a sharp increase in the boundary slope would result. Alternatively, for a region with high curvatures, we can choose the parameters for the auto-binary field so as to reflect locally this property. A parallel, iterative relaxation type algorithm arises naturally here, and is implemented to maximize the conditional likelihood $p(S|Y)$, or equivalently, the joint likelihood $p(Y|S)p(S)$. A sequential algorithm guaranteed to find a local maximum is as follows. Suppose at the nth iteration, the estimated binary field is S^n with value s^n_{ij} at pixel (i,j). Note that s_{ij} takes the values $(0,1)$ corresponding to texture data field classes $c = 1,2$ respectively. The different steps are:

1. Choose any pixel (i,j).
2. Keep s^n_{ij} as is or change it, whichever maximizes $p(S,Y)$. This provides a new estimate S^{n+1}. Hence

   $p(Y|S_{ij}, s_{ij} = k)\ P(S_{ij}, s_{ij} = k)$, $k = 0, 1$, must be computed.

   To carry out these computations we use

   $p(S_{ij}, s_{ij}) = p(s_{ij}|S_{ij})\ p(S_{ij})$.

   Since $p(S_{ij})$ is not a function of s_{ij}, only the conditional likelihoods $p(s_{ij} = k|S_{ij})$, $k = 0, 1$, must be computed. These are given in (6) and are simple to compute.
3. For the other computation, use

   $p(Y|S_{ij}, s_{ij} = k) =$

   $$p(y_{ij}|Y_{ij}, S_{ij}, s_{ij} = k) \cdot p(Y_{ij}|S_{ij}, s_{ij} = k),\ k = 0,1.$$

   Since the second factor is not a function of s_{ij}, all that need be computed is the first factor for $k = 0,1$.

4. Hence, choose $s_{ij}^{n+1} = 1$ or $s_{ij}^{n+1} = 0$ depending on whether the ratio (17) is greater than or less than 1.

$$p(y_{ij}|Y_{ij}, s_{ij}^{n+1} = 1, S_{ij}^{n})\, p(s_{ij}^{n+1} = 1|S_{ij}^{n})/ \\ p(y_{ij}|Y_{ij}, s_{ij}^{n+1} = 0, S_{ij}^{n})\, p(s_{ij}^{n+1} = 0|S_{ij}^{n}) \tag{17}$$

If some of the $y_{\ell,m}$ in the neighborhood system of y_{ij} are missing in Eq. (17), they are set equal to 0.

To carry out the next iteration, choose a new pixel (i',j') and repeat.

The parallel version of the above algorithm is based on dividing the entire data window into different subsets called codes. For example, for a 1st order auto-binary model the image data could be partitioned into two sets or codes as shown in Figure 6. For any specific pixel α, the conditional mean of the image function at that pixel α given the image at all other pixels will only depend on the 4 surrounding β pixels and not on any α pixels. Hence the algorithm described above could be applied to all the α pixels simultaneously. The next iteration would then involve working with all the β pixels. This algorithm will converge with fewer passes through the image than will the first algorithm described.

Note that the joint likelihood function is a multimodal function of S. The iterative algorithm is guaranteed to find a <u>local</u> maximum of the joint likelihood function in a finite number of iterations. If the binary field is used within a window as a smoother to the boundary estimated by the hierarchical algorithm, then only pixels adjacent to the hypothesized boundary at that time need be tested, rather than pixels interior to the hypothesized regions. In this sense, the algorithm would be related to the Ripple Filter introduced by Cooper, et al. [1,3].

4.2 <u>Proof that the Algorithm Converges</u>

At the end of each iteration the joint likelihood $p(S,Y)$ has increased or remained unchanged. For an $N \times N$ window, the joint likelihood, as a function of S, can take at most 2^{N^2} values. Because of this and the fact that the algorithm is sure that S^{n+1} differs from S^n only if the change strictly increases the value of the joint likelihoood function, it follows that the algorithm is guaranteed to find a local maximum of the joint likelihood function in a finite number of iterations.

4.3 <u>Example of the Form of the Classification Computation</u>

To illustrate explicitly the computations made during each iteration of the algorithm, suppose $p(y_{ij}|y_{\ell m}$ in region k) is

$$N(\mu(k) + \sum_{v \epsilon D_p} \beta_{r-v}[y_v - \mu(k)],\ \sigma^2(k))$$

with $r = (i,j)$, $v = (\ell,m)$.

Then the pixel texture-type classification function in its simplest form is to choose $s_r{}^{n+1} = 1$ or 0 at the $n+1$ st iteration in accordance with:

<u>If</u>
$$\begin{aligned} & -\ell n\sigma(1) - [1/2\sigma^2(1)]\{y_r-\mu(1) - \sum_{v \epsilon D_p} \beta^{(1)}_{r-v}[y_v - \mu(1)]\}^2 \\ & + \ell n\sigma(0) + [1/2\sigma^2(0)]\{y_r - \mu(0) - \sum_{v \epsilon D_p} \beta^{(0)}_{r-v}[y_v - \mu(0)]\}^2 \\ & + a + \sum_{v \epsilon D_p} b_{r-v} s^n_v \qquad (18) \end{aligned}$$

> 0, then $s_r^{n+1} = 1$; else, $s_r^{n+1} = 0$.

(This can be expressed more compactly by using matrix notation.)

4.4 A Modified Relaxation-Type Algorithm for Close-to-Optimum Segmentation

The algorithm in Section 4.1 generally converges to a local maximum of $p(Y,S)$. Segmentation errors take the form of classification errors in the vicinity of the true boundary and/or the occurence of one or more incorrectly classified sizeable regions. Let $\hat{s}_{ij}$ denote the value of s_{ij} produced by the segmentation algorithm. $\hat{s}_{ij}$ takes values of 1 and 0 in accordance with (18). It is usually the case that for many of the pixels (i,j) for which $\hat{s}_{ij}$ is incorrect, the likelihood ratio in (17)is close to 1. This suggests changing the values of all $\hat{s}_{ij}$ for which the likelihood ratio is close to 1, and using the resulting segmentation as an initial segmentation for rerunning the algorithm. The algorithm will converge to some new segmentation $\hat{\hat{S}}$. In almost all experiments that we have run, $\hat{\hat{S}}$ is closer to the true segmentation than is $\hat{S}$. The procedure can be repeated any number of times. It is possible for one of these perturbations to result in convergence to a less accurate segmentation. Consequently, it is desirable to use some measure of segmentation accuracy to determine when to accept a limit partition. The modified algorithm is as follows.

1. After convergence, run the following check at each pixel (i,j).

$$1 - \varepsilon \leq R_{ij} \leq 1 + \varepsilon \tag{19}$$

(where R_{ij} is the ratio in (17)).

2. If R_{ij} lies between the indicated bounds, change s_{ij}, otherwise keep it as it is.
3. After making all such changes throughout the image, use the resulting segmentation to run the parallel iterative algorithm until it converges again.

4. At each convergence $\tilde{S}$, compute the pseudo likelihood

$$\prod_{i,j} p(y_{ij}|\tilde{Y}_{ij},\tilde{S}_{ij})p(\tilde{s}_{ij}|\tilde{S}_{ij}).$$

(See Section 3.3.0 for the significance of this performance functional.) Repeat the algorithm until there is little change in a few successive pseudo-likelihoods. Keep that segmentation for which the pseudo-likelihood is a maximum. That segmentation is our estimate of the true segmentation. The reason the perturbation algorithm is so effective is that the segmentation resulting from the first convergent running of the algorithm is usually fairly good.

Two changes in the preceding algorithm could be considered. The first is to let ε decrease each time a new cycle is begun. The other is to not change the classification of all s_{ij} satisfying (19), but rather to change it with some probability, perhaps 0.5. This makes the algorithm more analogous to that used in other applications, and to the annealing methods explored recently, and earlier.

4.5 The Relaxation Algorithm Used for Seaming

If holding computation to an absolute minimum is not necessary, a simple convenient seaming procedure is to seam two (or four) adjacent windows by running the iterative parallel algorithm in the union of these windows. Since the algorithm begins with a close-to-maximum-likelihood segmentation, it goes through only a few iterations in order to converge. If the two (or four) windows contain the same two texture types, the hidden region modelling field used will be binary. If three texture types are involved, the region modelling field should be ternary. In a typical image such as Fig. 15c, if windows of modest size are used, the great majority contain at most two texture types; a small number of windows contain three texture types.

4.6 Segmentation of Images of Manufactured Objects

Smooth dull surfaces such as those of many manufactured objects result in picture functions that are spatially slowly varying and are well approximated over much of their extents by low degree polynomial functions. Such images are easily segmented using the adaptive hierarchical or iterative relaxation-type algorithms. One solution [23] is an extension of the solution to the problem of the segmentation of an image consisting of two or more regions of constant but unknown image intensity plus white Gaussian noise. See, e.g., Figs. 14a,b and the associated discussion in Section 6.0. In extending that solution, instead of the mean value of the picture function in region k being some a priori unknown constant $\mu(k)$, it might be a linear function taking value $\gamma_0(k) + \gamma_1(k)i + \gamma_2(k)j$ at pixel (i,j), where $\gamma_0(k)$, $\gamma_1(k)$, $\gamma_2(k)$ are a priori unknown constants. Or the mean value function might be a quadric function taking the value $\gamma_o(k) + \gamma_1(k)i + \gamma_2(k)j + \gamma_3(k)ij + \gamma_4(k)i^2 + \gamma_5(k)j^2$ at pixel (i,j). Hence, when using the adaptive hierarchical segmenter, the $\gamma_\ell(k)$'s could be first estimated using small image blocks and unsupervised learning. These estimates are used to start the hierarchical segmenter, which would adapt further during the segmentation procedure. Maximum likelihood or Bayesian parameter estimation is used during this stage. Adaptation can also be done when using the iterative segmenter.

5.0 Adaptation

An algorithm for segmenting textured images will be of no value unless it has the capability to estimate image model parameters during the segmentation process. This assertion is based on the observation that model parameters for a texture type will generally vary greatly from one image to another. The source of this parameter variability is variation in lighting, scale, and simply variability within a broad class such as tree foliage, grass, earth, roads, etc. (Within a single image, there generally is only slow spatial variation in the MRF parameters for a single texture type.) Consequently, either no a priori information is assumed concerning texture-type model parameters, or a priori distributions are assumed for these parameters, and these distributions are used in Bayesian-like parameter estimation during the segmentation process. In many instances the approximation that model parameters are constant over a texture-type region is satisfactory. In other instances, slow spatial variation must be assumed for model parameters. We briefly comment on how the first situation can be handled using our segmenters. An appropriate approach for the latter situation is slightly more complicated and involves modelling the spatial parameter variation for a texture-type model by an MRF. Practical incorporation of adaptation into segmentation algorithms is a substantial topic by itself and is treated in a subsequent paper. At the moment, we simply want to establish feasibility of solution.

Both the iterative-parallel and the hierarchical segmentation algorithms can function in an adaptive mode, but require some prior texture-type model parameter information. Three possible means of obtaining this information are as follows.

1. Use small windows, perhaps 16x16 or 32x32 in a single image to estimate model parameters. It is assumed here that the great majority of small windows of this size contain only one texture-type region. Effective parameter estimation is possible by maximizing the pseudo-likelihood function (see section 3.3.1). Parameter estimates from small windows in a texture-type region will cluster. Distributions for the various texture-type model parameters can then be obtained.
2. Obtain prior distributions for texture-type model parameters from previously segmented images.

3. Produce a first crude segmentation based on gray level or image features and image understanding. Use model parameters or parameter distributions estimated from this initial segmentation.

The hierarchical segmenter is ideally suited to adaptation since large region blocks are classified initially, and good parameter adaptation can be had by using these large blocks for parameter estimation as though each large block contained data from one texture-type. In a typical large block, most of the pixels will be from a region of the same texture-type. This is the approach exhibited in Figs. 9b, 11b. Parameter adaptation realized by incrementally updating parameter estimates following each iteration of the iterative segmentation algorithm is discussed in a forthcoming paper.

6.0 Experiments

The segmentation algorithms have been run on visible light, infrared, and artificially generated data. Generally, all of our algorithms work extremely well on stationary artificially generated MRF's, almost as well on stationary natural data, and less well but good on nonstationary natural data. The following examples illustrate these cases. Fig. 8a consists of two constant gray level squares in a background of some other constant gray level, plus white Gaussian noise. The signal-to-noise ratio (the difference in average values in the squares and background divided by the noise standard deviation) is 1. Segmentation by the iterative segmenter is carried out using an isotropic binary field having parameter values $a = -16.0$, $b_{\cdot,\cdot} = 1.5$. Fig. 8b is the initial segmentation accomplished by assigning a pixel to square or background depending on whether it is greater than or less than $(r_{obj} + r_{back})/2$, respectively. Here r_{obj} and r_{back} are the average picture function values in the object and the background, respectively. Following 21 passes through the image, the segmentation converged to is that in Fig. 8C. Fig. 9 is the infrared image of a tank and background to which has been added white Gaussian noise and illustrates the importance of parameter adaptation in the segmentation process. The same type of modelling used in Fig. 8 is appropriate here. The signal-to-noise ratio is 1.3, and a 2nd order binary MRF is used. Fig. 9a is the segmentation produced by the hierarchical segmenter using a ML (maximum likelihood) performance functional, and incorrect values for r_{obj}, r_{back}, σ. Starting with the same model parameters as in Fig. 9a but using the adaptive segmenter, results in the segmentation shown in Fig. 9b. Fig. 9c is the result of running this segmenter with parameters estimated using unsupervised learning on the window (see [2]) prior to segmentation.

In Fig. 10 we are seeing experiments with the iterative-parallel segmenter operating on artificially generated images. Two Gaussian textured regions are present, both with the same means and conditional variances, but otherwise different first order MRF's. There is a sinusoidal boundary separating the two regions. The upper region has strong correlation in the vertical direction, whereas the lower region has strong correlation in the horizontal direction. Model interaction parameters for the two <u>stationary</u> Gaussian fields are $\beta_{\cdot,\cdot}$ = .3, and -.1 for the vertical and horizontal directions in the upper image, and the permutation of these for the lower image. A second order binary S field was used with model parameters a = -4.6, $b_{\cdot,\cdot}$ = 2 in the horizontal and vertical directions and 0.3 in the diagonal directions. Note that since the mean values in the two texture-type regions are the same, it is impossible to do any meaningful segmentation based on thresholding the picture function. Spatial variation in the picture function must be exploited. Fig. 10a is the original image.

Fig. 10b is the initial segmentation based on use of the conditional likelihood of the picture function at a point given its surround under each of the two texture hypotheses. Fig. 10c is the segmentation resulting from the first convergence of the algorithm. The segmentation is then perturbed using the modified iterative-parallel segmenter (sec 4.1), and the algorithm run again. After four such cycles, the segmentation produced is that in Fig. 10d. Measured values of the pseudo-likelihood performance functional for the five segmentations at the convergence were -1.607e, -1.612e, -1.613e, -1.613e, -1.614e, respectively, where e is a constant. Note that the changes are small because the pseudo-likelihood function is for the segmentation of a whole window, and the number of pixel classification changes from one convergence to the next is small compared with the number of pixels in the window. But this performance functional is useful for deciding on whether a new convergent segmentation is better than an earlier one.

Again, Fig. 11 illustrates the all important adaptive segmentation. Fig. 11a is the segmentation of the image data in Fig. 10, but now using the hierarchical M.L. (maximum likelihood) segmenter with incorrect model parameters for the Gaussian MRF texture model. Fig. 11b is the result of starting off with the same incorrect parameters, but then running the segmenter in the adaptive mode. The resulting segmentation can now be smoothed using a few passes of the iterative segmenter with its binary region model. Hierarchical segmentation will also work well on such stationary data if the likelihood function without B determinant (see Eq. 5a), or the pseudo-likelihood function is used as a performance functional. These various performance functionals will give roughly the same segmentations down to block sizes of 8 x 8 and sometimes 4 x 4 pixels, but at the 2 x 2 pixel resolution use of the maximum likelihood performance functional results in more accurate segmentation. Hence, use of the determinant of B does make a difference! (see the discussion in Sec. 7.)

Nonstationary data provides the real challenge to segmentation algorithms based on MRF data models. Figs. 12a-12h illustrate a number of considerations in segmenting grass (upper) and earth (lower) regions in a portion of an outdoor visible light scene, using the hierarchical segmenter. Because maximum likelihood estimation of model parameters is computationally costly, we have experimented with the computationally less costly maximum pseudo-likelihood model parameter estimates. If the estimated parameter values result in B matrices (Eq. 5a) that are nonpositive semidefinite, which is the case in Figs. 12d, 12e, the Gaussian likelihood function cannot be used as a performance functional in the hierarchical segmenter. Consequently, we consider three performance functionals, namely, the L (likelihood), the L.W. (likelihood without the B determinant in Eq. 5a), and the P.L. (pseudo likelihood). In Figs. 12a-12e, parameter values are estimated prior to segmentation. Figs. 12a, 12b, 12c are hierarchical segmentations using the L., L.W., and P.L. performance functionals, respectively, and maximum likelihood parameter estimates. Second order Gaussian MRF models were used. In these figures, the image is divided into four windows, and segmenters are run independently in each. The maximum likelihood parameter estimates were constrained to be those for a <u>stationary</u> Gaussian model and were obtained by using asymptotic methods. Parameter estimates obtained are shown in the table. Figs. 12d, 12e are hierarchical segmentations using P.L. and L.W. performance functionals respectively, and pseudo-likelihood parameter estimates.

Estimator	texture-type region	σ^2	mean	vertical	horizontal	upper left lower right	lower left upper right
Maximum Likelihood	grass	32.0	158.0	.0339	.462	0	-.0036
	earth	136.3	150.9	.0142	.483	-.00164	-.00123
Maximum Pseudo Likelihood	grass	27.7	157.0	.263	.433	-.078	-.116
	earth	59.1	151.2	.451	.537	-.233	-.249

Table

Even though the image data is somewhat nonstationary, the combination of maximum likelihood parameter estimates and L. segmentation performance functional is best. In Figs. 12c, 12d and other experiments run with <u>nonstationary</u> image data, the P.L. segmentation performance functional results in segmentations that reasonably approximate the true boundary, but false boundaries are sometimes found as well. For such data, hierarchical L.W. segmentation usually reasonably approximates the true boundary when used with maximum likelihood parameter estimates, but sometimes completely misses the boundary, as in Fig. 12e, when used with maximum pseudo likelihood parameter estimates. The problem in this latter case may be caused by abnormally large positive values for the exponent in the likelihood-without-determinant for some erroneous segmentation because the B matrix is non positive semidefinite and therefore has at least one negative eigenvalue. Figs. 12f, 12g are the results of beginning with the segmentations in Figs. 12a and 12d, respectively, and seaming with the iterative segmenter. Notice that the effect of the seaming is to smooth out the estimates of the true boundaries and to remove some of the erroneously estimated regions. Finally, Fig. 12h is the classification of 8 x 8 pixel blocks in the image independently, using a P.L. classifier, as discussed in Sec. 3.3.0, with maximum pseudo likelihood parameter estimates. Note that almost all blocks are classified correctly, suggesting that with slight modification the hierarchical P.L. segmenter might be less prone to generating extraneous boundaries when the data is difficult to distinguish (16 x 16 pixel blocks are all classified correctly by the algorithm for this image.) Figure 13 is the hierarchical L. segmentation of two <u>nonstationary</u> artificially generated textures with sinusoidal boundary, based on use of maximum likelihood parameter estimates. The hierarchical P.L. segmentation based on maximum pseudo-likelihood parameter estimates was almost as good (there were no spurious segmentations), and this was true of the other segmenters as well.

Fig. 14a is an artificially generated image of a Lambertian can illuminated by a point source at infinity. Fig. 14b is the result of hierarchical segmentation. Of importance here is that the window to be segmented consists of portions of the images of the can top and can side at a location such that the image intensities of the two surfaces are the same in a small region in the center of the image. In other words, there is not a strong intensity discontinuity between regions! The image model used for the two surfaces is a constant plus noise for the can top, and a quadric polynomial plus noise for the can side. The segmentation algorithm is a trivial extension of our algorithms for segmenting models for which the image intensity over a region is constant plus white Gaussian noise.

In Fig. 15a we see an interesting image in which there is some deterministic geometric structure, largely in the house, but where the rest of the image is largely stochastic texture structure. We chose six texture-types to work with, the left tree foliage with roughly isotropic spatial variation (denoted tr_1), the foliage of the large tree on the right with greater spatial correlation in the vertical than in the horizontal direction (denoted tr_2), house roof shingles, skv, road, and grass. A set of MRF model parameters was estimated for each of these texture-types, from a small data window in each case. Then these models were used to generate slightly larger windows of artificial images, and the generated images were inserted in the picture close to the locations from which the data for estimating model parameters was taken, Fig. 15b. The artificially generated tr_1 texture-type image is a square window in the upper left corner of the picture. The other artificially generated square window images can be recognized in different locations in the picture. What is surprising is that the models appear to capture the texture structure amazingly well. All of the models used are 2nd order MRF's, except for the roof model which is 5th order. This higher order was necessary to capture the horizontal shingle structure because the angle involved is different from 0 degrees with respect to the horizontal axis for the picture. Fig. 15c illustrates the effectiveness of the pseudo-likelihood function (12a) as a recognizer of the texture-type seen within a window. Window data classifications were made independently of one another here. Notice that incorrect classifications were made in the dark area in the foliage of the left tree, and much of the sky region was incorrectly classified. The first set of errors is due to the fact that the model parameters for a texture-type vary (usually slowly) within an image, and this variation should be incorporated into the models used. The second set of errors is due to the occurrence of artifacts such as the images of telephone wires in the misclassified windows. Note that the bushes and a third small tree-foliage region on the right have been classified as tr_1 type regions. This is because models were not introduced for these texture-types, and the data seen looks very much like that in the tr_1 type region. Finally, note that two windows toward the right side of the grass region have been classified as road regions. Though it is difficult to tell from this image, these windows lie in a driveway region that looks very much like the road image texture. More windows in this region are not classified as road because these other windows contain grass image texture as well.

7.0 Physically Meaningful Models, Mathematical Correctness, and Computational Complexity

We comment on two important problems in this section.

1. Consider Eq. (17). The meaning of $p(y_{ij}|Y_{ij}, s_{ij}^{n+1} = 1, S_{ij}^{n})$ is that y_{ij} is conditioned only on those y_v in the neighbor set D_p in (5) for which $s_v^n = 1$. All other y_v in D_p are omitted; equivalently, they are set to 0. We shall refer to th y_v that are set to 0 as constituting missing data. The same situation arises with the hierarchical algorithm where the conditional likelihood of a block of data must be computed. This conditioning of y_{ij} by setting missing picture function values to 0 is not physically meaningful and also leads to probabilities of segmented images that do not constitute a consistent set of probabilities. (Note, the missing data problem also occurs at the four sides of a window, but since this window boundary is fixed and boundary effects die out away from the boundary, significant harmful effects on segmentation do not occur here.) Small improvement in segmentation accuracy can be made by correcting the problem. The solution is to treat $\{y_{ij}\}$ as though the texture-type 1 model is used to generate all the data in the window, and a subset of the data points is chosen, namely, those y_v for which $s_v = 1$. Then, the true conditional likelihood $p(y_{ij}|y_v$ for which $s_v = 1$, class 1) is computed. This conditional likelihood, which for reasons given is not (5), is Gaussian. The conditional variance of y_{ij} is at least as large as $\sigma^2(1)$ and lies between this and the marginal variance of y_{ij} for class 1. It can be computed approximately for various combinations of y_v present (see Appendix C, result 4). The true conditional mean of y_{ij} can be simply computed iteratively, as briefly described in Appendix C, result 5. Hence, with a modest amount of extra computation, a more physically meaningful probability measure, which is also consistent can be used for image modelling for the purpose of textured image segmentation. Of course, corresponding results apply when working with $s_{ij}^{n+1} = 0$.

The problem of having a consistent probability measure can be easily treated directly by designing a single MRF that incorporates both a region and a texture model [5,8]. Upon applying that formulation to our problem, the Gibbs potential would be

$$G = [Y_1-\mathcal{U}(1)]^t B(1) [Y_1-\mathcal{U}(1)] + [Y_0-\mathcal{U}(0)]^t B(0) [Y_0-\mathcal{U}(0)] + S^t BS. \quad (20)$$

Here, B(1) and B (0) are the inverse covariance matrices for the region 1 and the region 0 texture data vectors, respectively, and B is a matrix having values a along the main diagonal and b_{r-v} elsewhere. Note that the resulting p(Y,S) is

$$h \exp G \quad (21)$$

where h does not depend on the window partition for which p(Y,S) is computed. This likelihood is not the same as in (2a) using (5a) to specify $p(\{y_v\}: s_v=k | \text{class } k)$. The dependence of the likelihoods on the data Y_1, Y_0 is exactly the same in the two cases. However, whereas the likelihood which is the combination of 2a and 5a has a multiplicative factor $|B(k)|^{\frac{1}{2}}$ that is a function of the partition of the window, h in (21) is not a function of the partition. The multiplicative factor is substantial when the associated texture-type region comprises many pixels. It is $\sigma^{-1}(k)$ when Y_k consists of one pixel. It is possible to include additional quantities in (20) that results in a modified (21) being more like the combination of (2a) and (5a), but it is not clear that (21) can be modified to be close. For example, G can be modified to include $\sum \sigma^{-1}(s_v)$. This produces a somewhat strange MRF for modelling the image. However, use of this G in an iterative algorithm would result in exactly the same algorithm as in Eq. (18).

2. A second important problem is that the process defined by (5) may be stationary Gaussian, nonstationary Gaussian or nonGaussian, depending on the

β_{ij}'s. For example, as discussed earlier B in (5a) must be positive semidefinite for the MRF in an NxN window to be Gaussian. This restricts the β_{ij}'s. As $N \to \infty$, the β_{ij}'s become increasingly restricted, and in the limit satisfy the restrictions for a homogeneous Gaussian MRF. If B in (5a) is not positive semidefinite, the resulting exponential still defines a MRF if the exponential has finite integral. A sufficient condition for finiteness of the integral is that $|y_{ij}|$ be uniformly bounded above. This will always be the case for picture functions generated by real sensors.

8.0 Conclusions

MRF's appear to be very powerful for modelling textured images for the purpose of textured image segmentation and classification. New methods for image segmentation were presented in this paper. These algorithms are not computationally costly, and can be realized using parallel computer architectures, thus permitting real-time image segmentation. The algorithms are computationally simple. However, there are subtle, probably important, considerations to be aware of in using the approach presented. Further study of these issues is worthwhile. Fortunately, the MRF's have an interesting structure, and unlocking all of their secrets promises to be an interesting experience. For the moment, the reader's attention is directed to the following important considerations.

1. Since the image texture models used are Gaussian stochastic processes, do they provide any advantages over the use of Fourier analysis or finite impulse response filtering? The answer is a definite yes! The primary advantage is that Fourier analysis or finite impulse response filtering is applied over a rectangular window of image data. The window must be large enough to include at least a few spatial cycles of the picture function. Hence, the boundary

estimation accuracy achievable with this method is poor. On the other hand, segmentation of MRF models is very suitable to highly irregular boundaries. In the experiments we have run, boundary location estimation error usually lies between 0 and 3 pixels. Error analysis for the hierarchical algorithm is treated in a subsequent paper [22]. A second advantage of the MRF approach is that additional image model structure is easily incorporated. In this paper, we incorporated a texture region model. Boundary shape models can be incorporated. (We have found the use of boundary shape models to be of great use in earlier work [1,2]). A third advantage of the MRF's is that they are ideally suited to handling images where the texture parameters are a priori partially unknown or are spatially varying. These are the conditions usually encountered. In this paper, we have briefly exhibited an approach to treating the former case. In the latter case, the spatial variation can be appreciable. For example, in Fig.15 the texture in the dark portion of tree foliage type 1 is closer to the typical texture in the region of tree foliage type 2. Therefore, estimating the region occupied by a texture type should be based not on a fixed parameter model for a texture-type picture function, but rather on one where the texture model parameters at a point in a texture-type region are close in value to the model parameter values at nearby points in the region. In [23] a MRF is used to model parameter variation, and maximum likelihood estimation of these spatially varying parameters is shown to be an important ingredient of Bayesian textured image segmentation. Though the case of apriori fixed but partially unknown texture parameters is easy to handle using the Fourier analysis approach to image segmentation, it is not apparent how that approach could handle the case of spatially varying texture parameters. The one advantage of Fourier Analysis with respect to the MRF's is the ability to efficiently represent nonparametric spectra. However, texture segmentation is probably practical only for simple spectra having one or two modes. The MRF's should be good models for these, and indeed work well in practice.

2. It was shown that maximum likelihood segmentation of images of manufactured parts is possible with the same algorithms. Here, the picture function of a slowly curving 3-D surface is spatially slowly varying. Two approaches were taken to this problem. The first was the modeling of the image of a surface as a small piece of a MRF [13]. The MRF model parameters were treated as a priori. partially unknown. The second was to treat this image as a polynomial with a priori partially unknown parameter values plus white noise. In both cases, the partially unknown parameter values were estimated during image segmentation. More extensive treatment of adaptive segmentation using the polynomial model is given in a forthcoming paper.

3. Both segmentation algorithms presented are highly parallel , and will easily run in real time on the appropriate architectures (which can be built using present technology). The iterative relaxation algorithm has a very simple structure, and is well suited to use with general MRF models--- even those having non-constant parameters. Whereas the iterative algorithm will generally make only a small number of passes through an image, there are cases where it may make many passes when changing the classification of a region from an initial classification of one texture-type to a final classification of another texture-type. In these situations, the algorithm changes only a few pixels at the region boundary during each pass through the image. A way to overcome this is to use the iterative algorithm with 2x2 or 4x4 blocks or with blocks of decreasing size. Such a modification would run much faster and would be a hybrid of our hierarchical and relaxation algorithms. The hierarchical ripple filter [3] uses this philosophy.

The iterative algorithm is closely related to the annealing algorithms, but runs faster though it may not always be quite as accurate in finding a globally optimum image segmentation if the initial segmentation used in beginning the iterations is highly in error. However, what is most important is that the iterative algorithm will always work with nonstationary MRF models or with conditional Gaussian models when the parameters are such that the picture function is not quite Gaussian (see Sec. (6.0)) ; the annealing algorithms usually will not work under these conditions. But these conditions are the ones that are often prevalent in real image data!

The hierarchical algorithm achieves huge computational savings through certain divide and conquer type recursions (a logarithmic computational cost through use of a new simple ring structure) and also through organizing the bulk of the computation as computation on the data only. These data computations can then be combined with model parameters, at small computational cost, to realize texture segmentation. The complete computational cost benefit of the algorithm is realized only for fields with spatially constant parameters. However, the hierarchical algorithm can always be applied to nonconstant-parameter MRF's, and will always be effective in achieving high segmentation accuracy.

4. Three performance functionals are discussed and shown to be useful for the recognition of blocks of textured image or for hierarchical textured-image segmentation. These are the joint likelihood function with and without the determinants of the inverse covariance matrices, and the joint pseudo-likelihood function. The first functional usually produces the greatest accuracy, whereas the accuracies of the latter two are comparable. Examples of behavior are given in Sec. 6.0.

Acknowledgement:

This work was partially supported by Army Research Office grant #DAAG 29-81-K-0167, and National Science Foundation grants #ECS-8119676 and ECS-8404774. The authors are especially appreciative of the critical contributions of Judith Silverman. She did most of the programming and experimentation for the work reported. We also thank Magda Butnaru for her contribution to the programming and experimentation. Prof. A. Hanson, University of Massachusetts, Amherst, was kind enough to provide us with portions of the image data base developed by his group; the visible light data, Figs 12,15, came from that set.

References

1. D. B. Cooper, et al., "Stochastic Boundary Estimation and Object Recognition", Computer Graphics and Image Processing, April 1980, pp. 326-355.

2. D. B. Cooper and F. Sung, "Multiple-Window Parallel Adaptive Boundary Finding in Computer Vision", IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol. PAMI-5, No. 3, May 1983.

3. P. Schenker, et al., "Fast Adaptive Algorithms for Low Level Scene Analysis: The Parallel Hierarchical Ripple Filter", Conference Proceedings of the Fifth Inter. Symp. on Pattern Recognition, Miami, Dec., 1980.

4. L.L. Scharf and H. Elliot, "Aspects of dynamic programming in signal and image processing", IEEE Trans. Automatic Contr., vol AC-26, pp. 1018-1029, Oct. 1981.

5. H. Elliot, et al., "Image Segmentation Using Simple Markov Field Models", TR Ju81-DELENG-1, Colorado State Univ., Electrical Eng. Dept., June 1981.

6. C. W. Therrien, "Linear Filtering Models for Texture Classification and Segmentation", Proc. of 5th Int. Conf. on Pattern Recognition, Miami, Dec. 1980, pp.1132-1135.

7. H. Kaufman, et al., "Recursive Image Estimation: A Multiple Model Approach",Proc. 18th Conf. on Decision and Control, Fort Lauderdale, Florida, 12-14, Dec. 1979.

8. D. Geman and S. Geman, "Stochastic Relaxation, Gibbs Distribution and Bayesian Restoration of Images", Brown University., Division of Applied Math. T.R., September, 1983.

9. P. C. Chen and T. Pavlidis, "Image Segmentation as an Estimation Problem", Computer Graphics and Image Processing, Vol. 12, pp. 153-172, 1980.

10. A. Rosenfeld, R. A. Hummel, and S.W. Zucker, "Scene labeling by relaxation operations", IEEE Trans. Syst., Man., Cybern., Vol. SMC-6, pp. 420-453, June 1976.

11. O.D. Faugeras and M. Berthod, "Improving consistency and reducing ambiguity in stochastic labeling: an optimization approach", IEEE Trans. Pattern Anal. And Machine Intel., Vol. PAMI-3, pp. 412-424, July 1981.

12. F. S. Cohen, "Parallel Adaptive Hierarchical Algorithm for Textured Image Segmentation Using Noncausal Markovian Fields", Ph.D. Thesis, Division of Engineering, Brown University, Providence, RI., Aug. 1983.

13. F.S. Cohen and D.B. Cooper, "Real-Time Textured Image Segmentation Based on Non-Causal Markovian Random Field Models", Proc. of the 3rd Intl. Conf. on Robot Vision and Sensory Control, Cambridge, Mass., Nov. 1983; Proc. SPIE 449 , pp. 17-28 (1984).

14. F. Cohen, et al., "Simple Parallel Hierarchical and Relaxation Algorithms for Segmenting Textured Images Based on Noncausal Markovian Random Field Models," Proc. of the 7th Int. Conf. on Pattern Recognition, Montreal, Canada, July 1984.

15. J. Besag, "Spatial Interaction and the Statistical Analysis of Lattice Systems", Journal of the Royal Statistical Society, Series B., 36, 1974, pp. 192-236.

16. M.S. Bartlett, The Statistical Analysis of Spatial Pattern, Chapman and Hall, London, 1976.

17. J. Woods, "Two Dimensional Discrete Markovian Fields", IEEE Trans. on Information Theory, Vol. II-18, No. 2, March 1972, 232-240.

18. S. Ranganath and A. Jain, "Two Dimensional Linear Prediction Models Part I: SPECTRAL Factorization and Realization", T.R. SIPL-83-5, Dept. of Electrical Eng., Univ. of California at Davis, May 1983.

19. M. Ekstrom and J. Woods, "Two Dimensional Spectral Factorization with Applications in Recursive Digital Filtering", IEEE Trans. Acoust., Speech, Signal Processing, Vol. ASSP-24, #2, pp. 115-128, April 1976.

20. R.J. Cross, "Markov Random Fields Texture Models", TR 80-02, Dept. of Computer Science, College of Engineering, Michigan State University, 1980.

21. R.L. Kashyap, et al., "Estimation and Choice of Neighbors in Spatial Interaction Models of Images", IEEE Trans. on Information Theory, Jan. 1983, pp. 60-72.

22. Cohen paper in preparation.

23. Cooper, et al., paper in preparation.

24. M. Loeve, <u>Probability Theory</u>, D. van Nostrand, 1960, p. 228.

25. G.E. Hinton and T.J. Sejnowski, "Optimal Perceptual Inference" Proceedings IEEE Comp. Soc. Conf. on Comp. Vision and Pattern Recog., Washington, D.C., June 19-23, 1983, pp 448-453. Also, talk at Brown Univ., April, 1984.

Appendix A

Proof of Result 2.

Under the free boundary condition,

$p(X,Y) = \text{constant}\cdot\exp\{-(1/2\sigma^2)[X^tB_XX + 2X^tB_{XY}Y + Y^tB_YY]\}$

From this it can be seen that $p(X|Y)$ is Gaussian with covariance matrix B_X^{-1} and mean vector $-B_X^{-1}B_{XY}Y$. Hence,

$$\begin{aligned} p(X|Y) &= |B_X|^{\frac{1}{2}}(2\pi\sigma^2)^{-N_X/2}\cdot\exp\{-(1/2\sigma^2)[(X + B_X^{-1}B_{XY}Y)^tB_X(X + B_X^{-1}B_{XY}Y)]\} \\ &= |B_X|^{\frac{1}{2}}(2\pi\sigma^2)^{-N_X/2}\cdot\exp\{-(1/2\sigma^2)[X^tB_XX + 2X^tB_{XY}Y + Y^tB_{XY}^tB_X^{-1}B_{XY}Y]\} \end{aligned}$$

Appendix B

Theorem

Consider C homogeneous Gaussian MRF texture models; the covariance matrix for every set of pixels in a texture-type region is nonsingular. Let $\ell(Y|k)$ denote the discriminant $(1/N^2)\,\ln[\prod_{(i,j)\in\Omega} p(y_{ij}|Y_{ij},k)]$, $k = 1,2,\ldots,C$, for the kth texture-type in an NxN pixel window. The texture recognizer based on choosing the texture type having the largest discriminant incurs a probability of misclassification that goes to 0 w.p.1 as $N \to \infty$.

Proof

Assume that the texture-type under which the data was generated is k_o.

$(Y|k) =$

$$= -(\tfrac{1}{2})\ln[2\pi\sigma^2(k)] - 1/N^2 2\sigma^2(k) \sum_{\substack{(i,j)\\ \in\Omega}} \{y_{ij} - \mu(k) - \sum_{\substack{(\ell,m)\\ \in D_p}} \beta_{\ell-i,m-j}(k)\,[y_{\ell m} - \mu(k)]\}^2 \qquad \text{(B1)}$$

is the sample mean of approximately N^2 identically distributed, partially correlated r.v.'s

$z_{ij}(k) =$

$$= -(\tfrac{1}{2})\ln[2\pi\sigma^2(k)] - [1/2\sigma^2(k)]\{y_{ij} - \mu(k) - \sum_{\substack{(\ell,m)\\ \in D_p}} \beta_{\ell-i,m-j}(k)\,[y_{\ell m} - \mu(k)]\}^2 \qquad \text{(B2)}$$

Since $\{y_{ij}\}$ is homogeneous, so is (B2) for all k. Because $\{y_{ij}\}$ is a nonsingular homogeneous MRF, both its autocorrelation function and that of (B2) fall off exponentially for large distance from the origin. Using this and the Borel Cantelli Lemma [24], it is easy to prove a strong law of large numbers, specifically that

$$\ell(Y|k) \xrightarrow[\text{W.P.1}]{} E[z_{ij}(k)|k_o] \quad \text{as } N \to \infty, \tag{B3}$$

where the expection is with respect to the measure for Y.

Claim:

$$E[z_{ij}(k_o)|k_o] > E[z_{ij}(k)|k_o] \quad \text{for } k \neq k_o. \tag{B4}$$

This comes from a standard information theory argument as follows.

$$E[z_{ij}(k)|k] =$$

$$= \int \ell n \left\{ p(y_{ij}|Y_{ij},k_o)\left[1 - \frac{p(y_{ij}|Y_{ij},k_o) - p(y_{ij}|Y_{ij},k)}{p(y_{ij}|Y_{ij},k_o)}\right]\right\} \cdot$$

$$\cdot\ p(y_{ij}|Y_{ij},k_o)p(Y_{ij}|k_o)dy_{ij}dY_{ij}$$

$$\leq \int [\ell n p(y_{ij}|Y_{ij},k_o)]p(y_{ij}|Y_{ij},k_o)p(Y_{ij}|k_o)dy_{ij}dY_{ij} \tag{B5}$$

$$- \quad [p(y_{ij}|Y_{ij},k_o) - p(y_{ij}|Y_{ij},k)]p(Y_{ij}|k_o)dy_{ij}dY_{ij}$$

$$= E[z_{ij}(k_o)|k_o],$$

with equality for $p(y_{ij}|Y_{ij},k) = p(y_{ij}|Y_{ij},k_o)$ for all y_{ij}, Y_{ij}. Since $p(y_{ij}|Y_{ij}, k)$ is an exponential function in y_{ij} and Y_{ij}, because of the nonsingularity of the process it follows that there is equality <u>only</u> if $k=k_o$. Inequality (B5) comes from use of the inequality

$$\ell n(1 + \varepsilon) \leq \varepsilon \quad \text{for} -\infty < \varepsilon < \infty.$$

Because of the preceding inequality and (B3),

$\ell(Y|k_o)/\ell(Y|k) \xrightarrow[\text{W.P.1}]{} a$ as $N\to\infty$, $k\epsilon\{1,2,\ldots,C\}$ and $k \neq k_o$, where a is a constant strictly greater than 1. Thus, probability of correct classification converges to 1 with probability 1 as N goes to infinity.

The proof can be extended to nonGaussian MRF's and to some nonstationary ones. A simple procedure is to consider the sequence L_N, where L_N takes the value 1 if $[\sum_{(i,j)\,\epsilon\,\Omega} p(y_{ij}|Y_{ij},k_o)/p(y_{ij}|Y_{ij},k)] < 1$, and is 0 otherwise.

Note that $k \neq k_o$ in the inequality. Then the idea is to prove that as $N\to\infty$, L_N takes the value 1 only finitely often. But this will be the case, by the Borel Cantelli Lemma, if $(\sum_{N=2}^{\infty} p\{L_n = 1\})<\infty$. Hence, this proof depends on showing for the MRF model under consideration that the preceding sum of probabilities is finite.

Appendix C

Some useful likelihood functions

Let an NxN pixel window be partitioned into three regions, and let the picture function at the pixels in these regions be components of the column vectors W, X, Y. Let $Z = (X^tY^t)^t$. The picture function over the window is a constant parameter Gaussian MRF. Then the following results are already derived in the paper or can be derived using similar techniques. Let the field have 0 mean value.

1. $$p(W,Z) = (2\pi\sigma^2)^{-N^2/2}\ |B|^{\frac{1}{2}}$$
$$\cdot \exp\ \{-(1/2\sigma^2)\ [W^tB_WW + Z^tB_ZZ + 2\ W^tB_{WZ}Z]\}$$

where B, B_W, B_Z are the <u>within</u> interaction matrices for the whole window, W, and Z, respectively; and B_{WZ} is the <u>between</u> interaction matrix for W and Z. See Secs. 2.2.1 and 3.2.1.

2. $p(Z|W) \sim N(-B_Z^{-1}B_{ZW}W, \sigma^2 B_Z^{-1})$

   (See Section 3.2.1)

3. (i) $p(Z) \sim N(0, \sigma^2[B_Z - B_{ZW}B_W^{-1}B_{WZ}]^{-1})$

This result follows from $p(W,Z) = p(W|Z)p(Z)$ and use of results 1 and 2. Note that the covariance matrix for Z is "bigger" than $(B_Z)^{-1}$, because $(B_Z)^{-1}$ is the <u>conditional</u> covariance matrix for Z given that the surrounding neighbors have value 0.

(ii) $$p(Z) = cp(\hat{W},Z)$$

where c is a suitable normalizing constant, and $\hat{W}$ is the value of W that maximizes the joint likelihood $p(W,Z)$. This result is discussed in [12; pp. 55,56] and a simple iterative relaxation algorithm for computing $\hat{W}$ is discussed in [13; Appendix], and also described as result 5 in this Appendix of this paper.

4. $p(Y|W) \sim N(\,.\, , \sigma^2[B_Y - B_{YX}B_X^{-1}\, B_{XY}]^{-1})$

This result follows from $p(X,Y|W) \sim N(.,B_Z^{-1})$ obtained from use of result 2, and then the application of result 2 to $p(X',Y')$ where Z' is $Z|W$. The fact that $\mathrm{cov}[Y|W]$ does not depend on B_W or B_{YW} makes sense, since the covariance matrix for Y when Z is conditioned on the free boundary condition at its border is exactly the same as when conditioning on a combination of the free boundary condition and on W where W borders on Z.

5. We describe a simple algorithm for computing $E[Y|W]$ when W, X, Y, Z are as previously described. Note that $E[Z|W] = [(E[X|W])^t, (E[Y|W])^t]^t$, so that $E[Y|W]$ is the lower vector in $E[Z|W]$. From $p(Z,W) = p(Z|W)p(W)$, it is seen that $E[Z|W]$ is the value of Z for which $p(Z,W)$ is a maximum, i.e., it is the maximum likelihood estimate of Z. But $p(Z,W)$ is a quadric form in Z with positive semidefinite matrix. Assume the matrix is positive definite. Hence, $p(Z,W)$ has only one maximum as a function of Z, and any one of many iterative algorithms may be used to compute this value of Z. We use a relaxation algorithm. Consider updating the estimate z^n_{ij} of the component of Z at pixel (i,j) found at the nth stage. Then z^{n+1}_{ij}, the estimate at the n+1 stage, is computed as

$$z^{n+1}_{ij} = \mu + \sum_{v \epsilon D_p} \beta_{r-v}(t^n_v - \mu)$$

where this is the expression for the conditional mean in Eq. (5), and t^n_v is a component of either Z^n or W, depending on v. As with the relaxation algorithm in Sec. 4.1, estimates at all locations in a code can be made simultaneously.

5	4	3	4	5
4	2	1	2	4
3	1	X	1	3
4	2	1	2	4
5	4	3	4	5

Figure 1

0	0	0
1	X	0
1	1	0

Figure 2

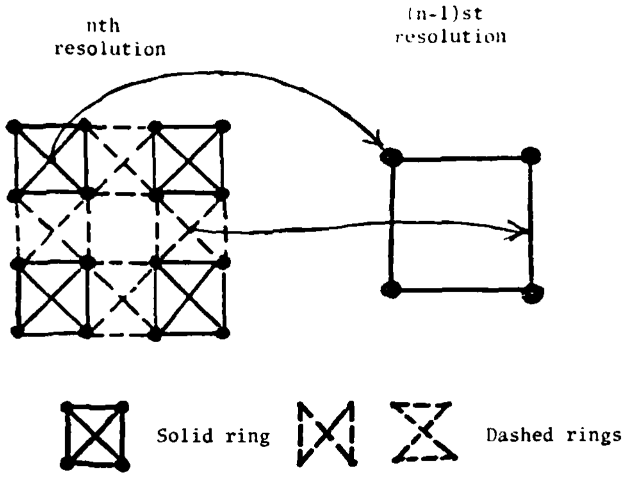


Figure 4

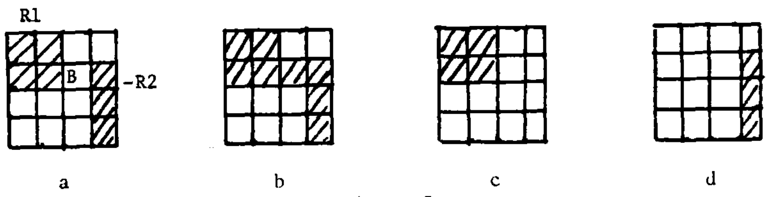
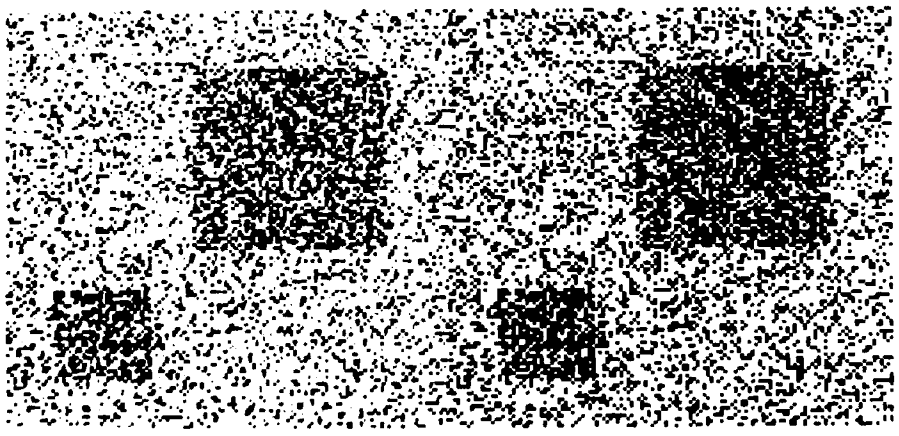


Figure 3



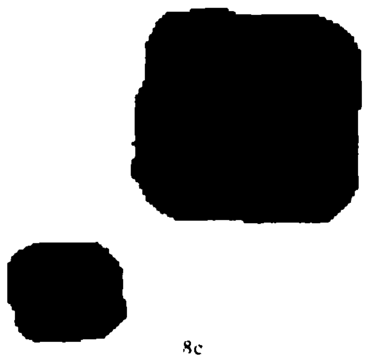


Figure 8

a= -16.0
b(.,.)=1.5
α= 0.25
Means = 20 & 20
Std. dev. = 40

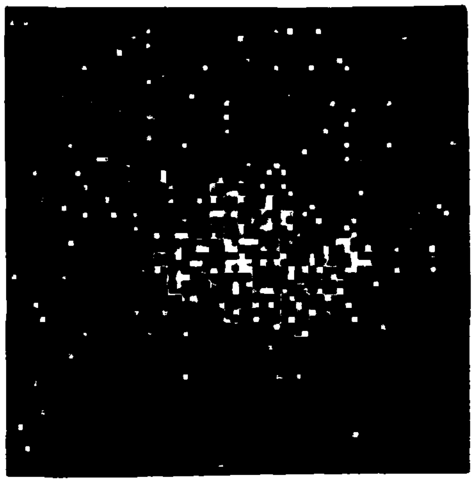


Figure 9a

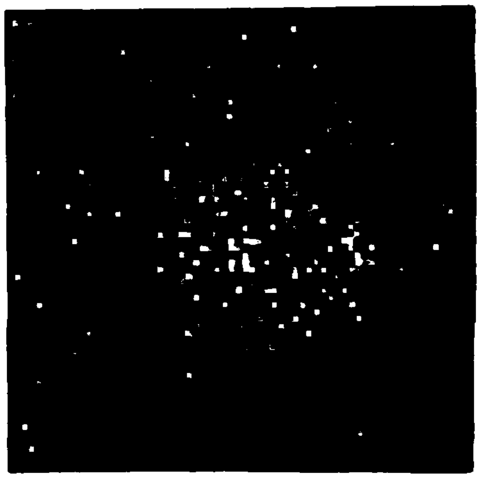


Figure 9b

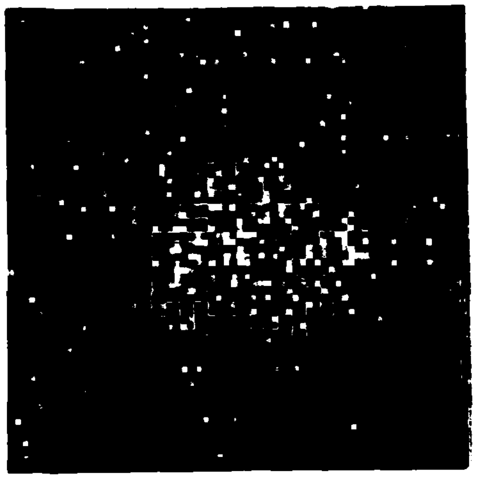


Figure 9c

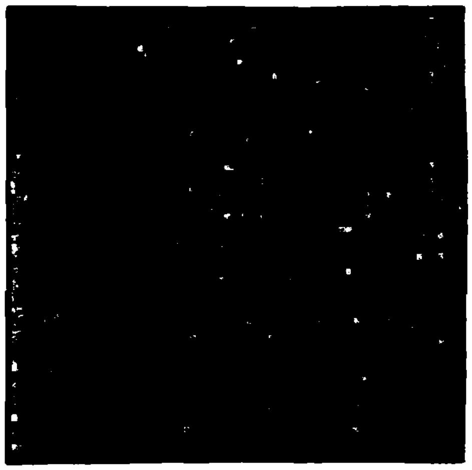


Figure 10a

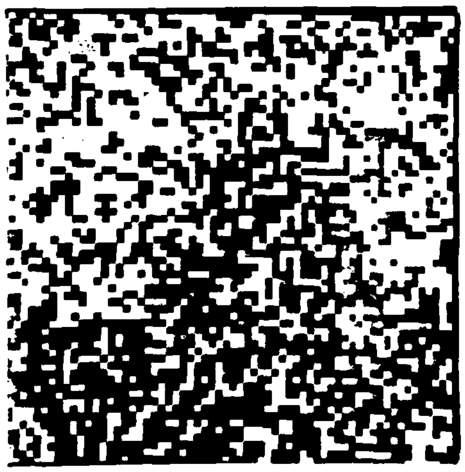


Figure 10b

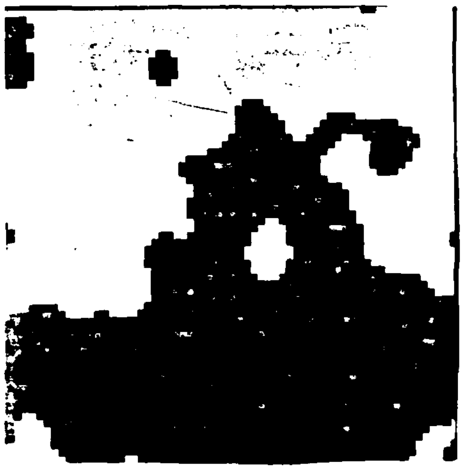


Figure 10c

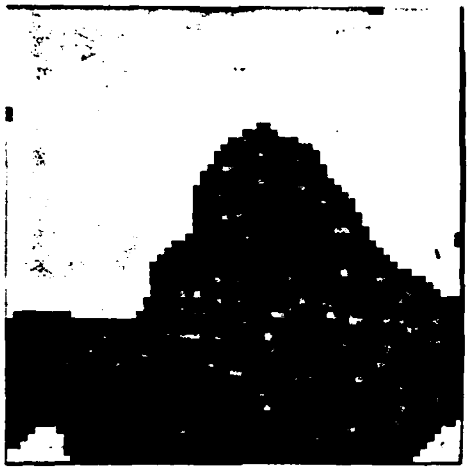


Figure 10d

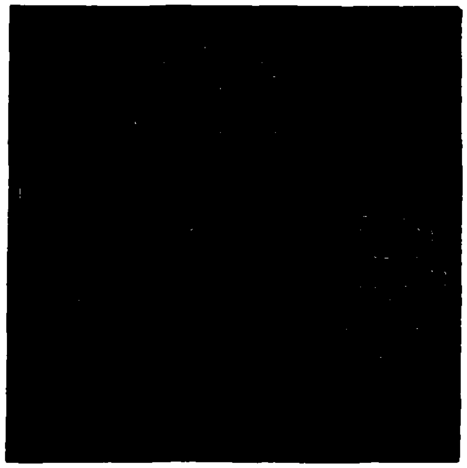


Figure 11a

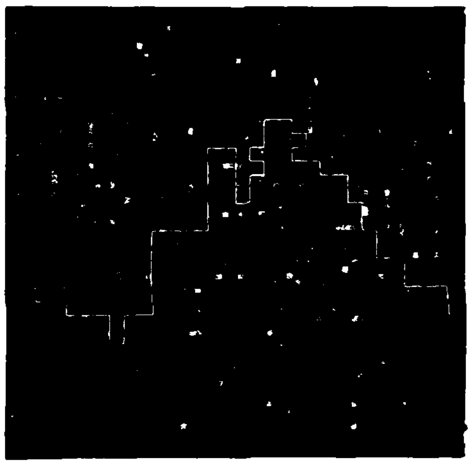


Figure 11b

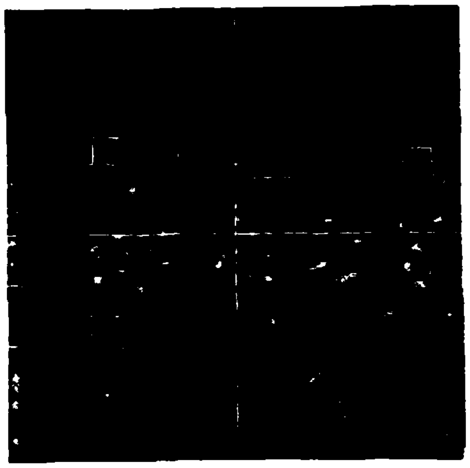


Figure 12a

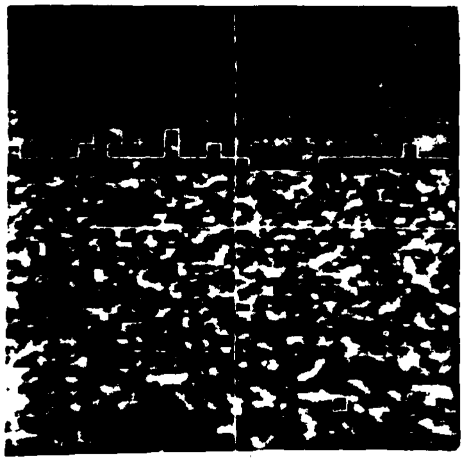


Figure 12b

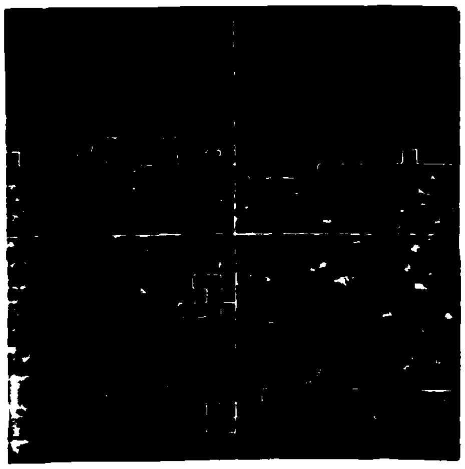


Figure 12c

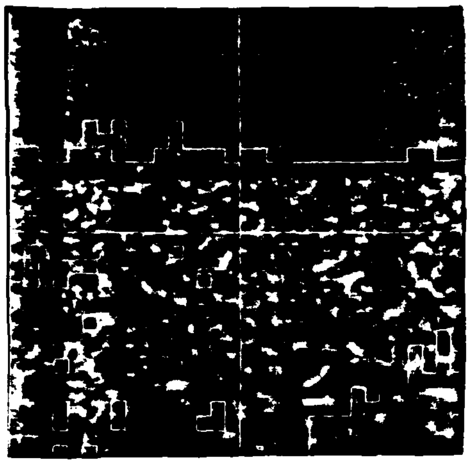


Figure 12d

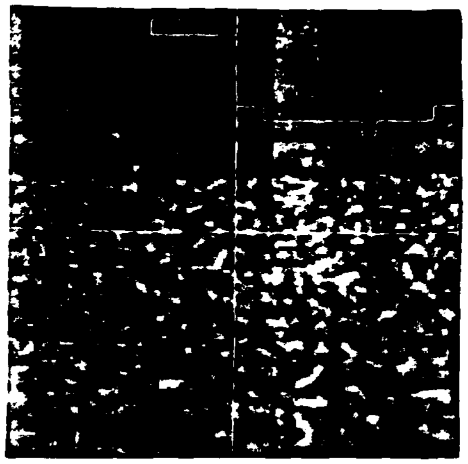


Figure 12e

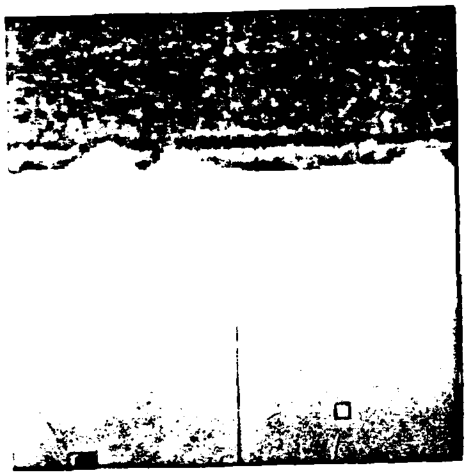


Figure 12f

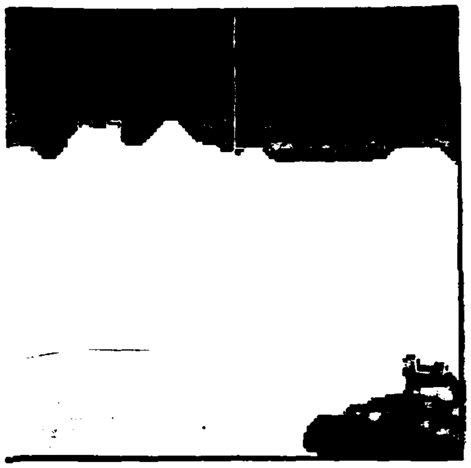


Figure 12g

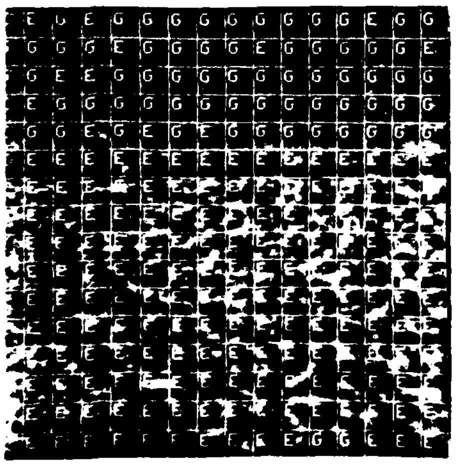


Figure 12h

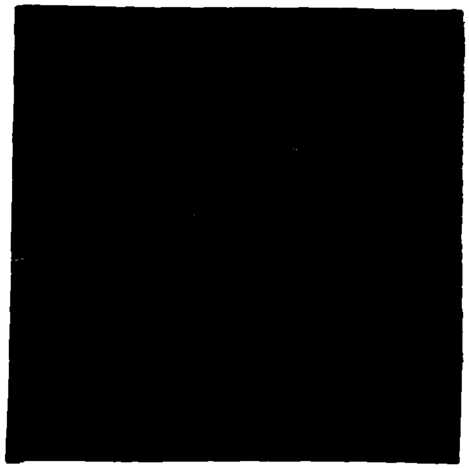


Figure 13

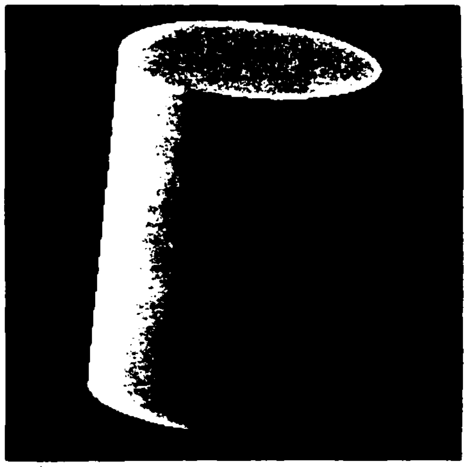


Figure 14a

Figure 14b

Figure 15a

Figure 15b

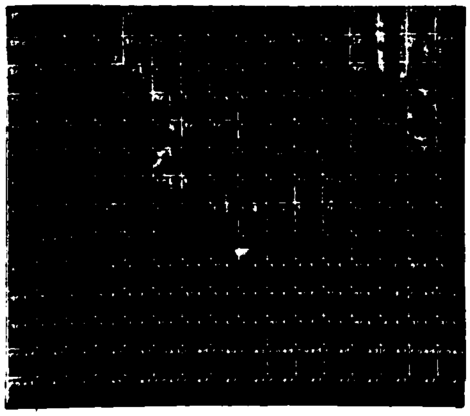


Figure 15 c